Disjunctive Argumentation Semantics (DAS) for Reasoning Over Distributed Uncertain Knowledge

by
Benson, Ng Hin Kwong

香港中文大學
THE CHINESE UNIVERSITY OF HONG KONG

Submitted to the Department of Systems Engineering &
Engineering Management
and the Faculty of Engineering of
The Chinese University of Hong Kong
in partial fulfilment of the requirement for the degree of
Master of Philosophy

June 1998

Time present and time past
are both perhaps present in time future.
and time future contained in time past.
If all time is eternally present,
all time is unredeemable.

<< Burnt Norton >>, T.S.Eliot

基於〝選言辯論語義〞的分佈不確定知識推理

摘要

在公開領域中基於分佈知識的應用，其所依賴的常識性信息是不完整的、不確定的和不一致的。不確定形知識或資訊的不完整性、不明確性、不一致性並非相互獨立，而是互相聯系的。爲了抽取有用的結論，上述的三方面問題必須融合貫通地解決。據我們所知，目前還沒有一種集成性的理論架構可以處理以上的問題。在另一相關領域—法理推理體系中，〝辯論〞是一種廣泛被應用的方法學。進年的研究顯示〝辯論〞十分適合用於分散式不確定型常識性推理的應用。這揭示了解決以上問題的新可能性。

在本論文中，我們運用辯論理論分析不確定型知識所引起的衝突。我們提出兩個分別名爲〝選言辯論語義Ⅰ〞及〝選言辯論語義Ⅱ〞的集成框架。我們給出〝選言辯論語義Ⅰ〞及〝選言辯論語義Ⅱ〞不單能夠表達不完整的、不明確的和不一致的知識，更能分析及判別這些知識。我們透過一組充分的標準測試和一個偵探推理問題展示我們所提方法的可用性與實用性。從這些測試中，這兩個集成框架都能因應不同的須要而作出適當的調整。

Abstract

Distributed knowledge based applications in open domain rely on common sense information which is incomplete, indefinite and inconsistent. Incompleteness, indefiniteness and inconsistencies are not orthogonal in uncertain knowledge/information and are closely inter-related. To draw useful conclusions from uncertain information, one must address all three aspects in a holistic manner. To our best knowledge, there is no existing reasoning framework which adopts an integrated approach on this problem in artificial intelligence. In another camp, argumentation is a widely used methodology in legal reasoning. Recently, it has been shown to be useful for common sense reasoning over distributed uncertain knowledge. This sheds new light in solving the above problem.

In this thesis, we analyze conflicts incurred by uncertain information in the light of argumentation theory. We present two integrated frameworks, namely Disjunctive Argumentation Semantics I (DAS-I) and Disjunctive Argumentation Semantics II (DAS-II). We show that DAS-I and DAS-II cannot only model incomplete, indefinite and inconsistent information but also analyze and resolve them. Furthermore, we show that our approach is practical as well as useful by extensive evaluations using a set of well-known benchmark problems and a detective reasoning problem. We show that both frameworks support multiple views can be defined on a set of distributed knowledge and conclusions could be made subjectively according to the different situations.

Acknowledgement

Thanks to my dad and mom. Without them, I am nothing. I am gifted to have an excellent family. My parents support whatever decisions I made though they know little about what I am doing. My two sisters give me ceaseless support. My girlfriend Dilys Hung helped proof-read my first rejected paper in despairing time. Encouragement from my farlong friends Mr. Derek Ip and Mr. Grand Sze is invaluable. All these people have much more faith and confidence on me than myself. I am indebt to their endless love and care.

Thanks are due to my two co-supervisors. I am grateful to Professor Boon-Toh Low who gives me the freedom, space and time to pursuit what I believe. I would like to dedicate my sincere thanks to Professor Kam-Fai Wong who also supervised my undergraduate thesis. Throughout these years, he is not only my mentor but also my good friend. His dedication to perfection and ceaseless hardworking style remind me what a scholar should look like. He is always my teacher.

Another great person I met in the University is Mr. Sai-Kee Wong. It was from him I first learnt that "Engineering is an art when you do it in the right way.". My philosophy on the engineering side took shape while I was attending his tutorial classes and in correspondance with him. I am also grateful to have enrolled in Professor Man Hon Wong's class which gives me lots of inspirations in theoretical studies.

I am thankful to my colleagues in the Information Systems Laboratory. I learn the real elegance of computer science from Mr. Wai-Ip Lam who is a world-class scientist in my view. I am grateful to Mr. Ka-Ho Ma who has always been so tolerable to my poor tennis skill and my mean character; Mr. Chi-Yin Wong and Mr. Chi-Wang Yue who go swimming with me and has brought much happiness to many boring working days. Mr. Xiao Dong Chen, Mr. Edmund Chiu and Mr. Wai Kwong Leung who keep me excercising my muscle in leisure time.

I find it enjoyable and meaningful to have taught bright and smart students in the past two years. Nothing can be more rewarding than the transfer of my knowledge and beliefs to them. Last but not least, I would like to dedicate my achievements, if any, to the technicians and teaching staffs of the Department, and all those I have not mentioned here but who love me.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

> ... formalizing knowledge and mechanizing reasoning, both commonsense and refined expertise, in all areas of human endeavor ...
>
> *Strategic Directions in Artificial Intelligence*
> Jon Doyle, Thomas Dean et. al.

Commonsense reasoning capability is indispensable for artificial intelligence (AI) applications. Knowledge involved in these applications is usually uncertain. They are incomplete, indefinite and inconsistent.

Knowledge is incomplete as we cannot represent all things in a finite system. Moreover, we want to make use of the unrepresented part, e.g. "If John is not shown to be guilty, John is innocent." There are infinitely many facts for showing "John is guilty" but we cannot represent them all.

Knowledge is usually indefinite as there are facts undecidable at the moment, e.g. "If John is betting on a horse, John will either win or loss.". Problems that have more than one solutions usually contained a significant amount of indefinite information. In real life, we often tackle this type of problems by reasoning different cases and their possible outcomes as the reference model.

Knowledge could also be inconsistent. Opinions from different experts can be contradictory. Opinions from a single individual can also be contradictory, e.g. "If Mary is a penguin, Mary is a bird. If Mary is a bird, Mary can fly." and "If Mary is a penguin, Mary cannot fly. ". Sometimes we know how to resolve inconsistencies and we may simply suppress inconsistent information for other time.

In AI, it has been shown that classical methodologies like propositional logic have difficulties to handle the above problems. Simply abandoning uncertain knowledge is unrealistic, particular in reasoning applications. Since 1960, researchers studied how to model and overcome this predicament. Sophisticated analyses have been performed

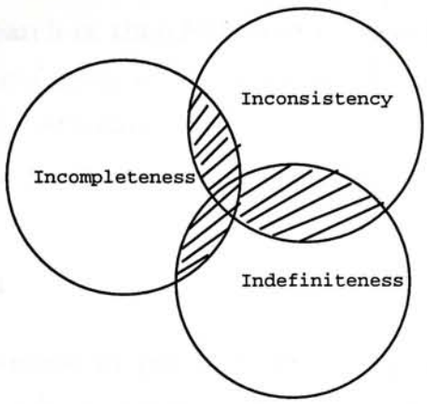

Figure 1.1: Relationships between incompleteness, indefiniteness and inconsistencies

and solutions have been devised for individual uncertain information type. It seems that direct combination of these solutions could give a complete answer. However, the metaphor of divide-and-conquer does not apply here. The problems lie in the assumption behind the solutions of the individual parts, i.e. incompleteness, indefiniteness and inconsistencies were assumed independent. But in-depth analysis in human reasoning shows that this is not the case. Incompleteness, indefiniteness, inconsistencies are inter-related, see Figure 1.1. Therefore, a conclusion drawn by considering only one aspect of uncertainty is unacceptable and could lead to fatal consequences. Consider the following example.

1. `If John is not shown to be guilty, John is innocent.`
2. `If John is not shown to be dishonest, John is not innocent.`
3. `If John is not innocent, John is guilty.`

Sentence 1 use incompleteness modeling technique (i.e. `not shown`). Sentences 2 and 3 together show that John is guilty and not innocent. The later result refutes the incomplete assumption in Sentence 1. Sentence 1 is inconsistent with Sentences 2 and 3. Suppose we directly combine both incomplete and inconsistent information handling frameworks and further assume that the inconsistencies resolving technique, specificity[1] is used. Sentence 2 and 1 are equally specific on `innocent`. There is no way to determine which one is preferred. However, everyone would deem Sentence 2 and 3 together a better argument than Sentence 1 alone as assumption of Sentence 1 (i.e. non-existence of `guilty`) is refuted. On the contrary, assumption of Sentence 2 and 3 (i.e. non-existence of `dishonest`) is not refuted. This shows that direct combination is unacceptable.

[1]We shall discuss the details of specificity in Section 2.2.2.

The objective of our research is, therefore, to overcome the aforesaid problem which has been undermining the usefulness of existing commonsense reasoning systems. Our goal is to design a reasoning methodology which can handle indefinite, incomplete and inconsistent information holistically (i.e. in an integrated fashion).

1.1 Our approach

Consider the example mentioned in previous section again. In the analysis, we use the concept of "arguments" which is commonly used by human beings. The example shows that reasoning with "arguments" is more suitable to analyze relationships between incomplete, indefinite and inconsistent information. By thinking knowledge as arguments, we can identify different opposing opinions in knowledge, which are technically referred to as conflicts, e.g. inconsistencies and refuting assumptions.

Argumentation is a newly revival methodology dealing with conflicts. Different from approaches in classical logics, argumentation is controversy-oriented. It concentrates on modeling relationships between conflicts and resolving conflicts rather than simply avoiding conflicts in classical logics. As shown in substantive research efforts [Kowalski and Toni, 1996, Lin and Shoham, 1989, Verheij, 1996], argumentation is also good at unifying different frameworks and methodologies. Thus, we propose an argumentation oriented approach to tackle incompleteness, indefiniteness and inconsistencies as a whole.

Rather than starting from scratch, we adopt Henry Prakken's strict argumentation framework as our basis as it provides an integrated method to incomplete and inconsistent information handling. However, there is no notion of indefiniteness in Prakken's framework. This motivated us to extend it. In addition, we analyze relationships between incompleteness and indefiniteness, as well as inconsistencies and indefiniteness. We propose an integrated framework, Disjunctive Argumentation Semantics I (DAS-I) to tackle the problem. The design rationale of DAS-I is to be as simple as possible. As such DAS-I does not support "reasoning about cases". To overcome this deficiency, we propose a second framework, namely Disjunctive Argumentation Semantics II (DAS-II).

Our main contribution in this thesis is the proposal of DAS-I and DAS-II. To our best knowledge, DAS-I and DAS-II are the only two attempts to merge disjunctive logic and argumentation in a distributed setting. They are also novel in identifying new kinds of conflicts between distributed knowledge bases, namely thinning and unbalanced rebut.

1.2 Organization of the thesis

This thesis is organized as follow: In Chapter 2, we review the basics of logic programming and basic requirements of practical reasoning frameworks. This is followed by a description of the background theories of argumentation, a newly revival approach to reasoning, in Chapter 3. We show that there is a variant of argumentation which meets all basic requirements set out in Chapter 2. We then present our two formulations of disjunctive argumentation semantics in chapters 4 and 5. In Chapter 6, we evaluate our formulations using well-known test cases and an interesting example. After that, we summarize our work and highlight our research contributions in Chapter 7.

Chapter 2

Logic Programming

Our work on disjunctive argumentation is built on top of logic programming. Logic programming was chosen because its goal of devising a computation mechanism for automated reasoning coincides with the objective of our disjunctive argumentation framework.

Logic programming began with the important work of Robert Kowalski [Kowalski, 1974]. In 1972, he formulated the important interpretation of clausal form logic as a programming langauge. His work was based on previous results in automated theorem proving, Alfred Tarski's semantic theory [Tarski, 1956], Skolem's model-preserving transformation, Jacques Herbrand's theorem [Herbrand, 1967] on finite characterization of inconsistency, Davis and Putnam's clausal form logic [Davis and Putman, 1960], and Alan Robinson's resolution procedure [Robinson, 1965]. Based on them, Kowalski presented his seminal paper [Kowalski, 1974] on procedural reading of predicate logic programming which pioneered the field of logic programming. Since then numerous researchers studied both computational and theoretical asepcts of logic programs. Improvements were made on the following areas:

- Negation-as-finite-failure is proposed for reasoning with incomplete information.
- Paraconsistency and prioritized reasoning are proposed for reasoning with inconsistent information.
- Disjunctive logic programming is proposed for reasoning with indefinite information.

In this chapter, we outline the basics of logic programming as well as the aforementioned improvements. In Section 2.1, we describe the properties and problems of logic programming in Horn clauses, a subset of First Order Logic (FOL) [1]. We then discuss

[1] In this thesis, classical notions of First Order Logic will be heavily used. Readers who do not familiar with First Order Logic are advised to consult Appendix A.

solutions to the problems in Section 2.2. In Section 2.3, we render the difficulties when incomplete, inconsistent and indefinite information co-exist. In Section 2.4 and 2.5, we discuss two integrated attempts, stable semantics and well-founded semantics for solving parts of the problem in reasoning under incomplete, inconsitent and indefinite information.

2.1 Logic programming in Horn clauses

Kowalski's framework [Kowalski, 1974] is defined on a special class of clauses in which every clause has at most one non-negative atomic literal. This class of clauses is commonly known as Horn clauses in memoir of Alfred Horn. Horn clause logic was chosen by Kowalski for its goal-directness property, which gives exceptional computational advantage, and model uniqueness property, which gives a clear cut semantical definition.

A Horn clause is usually represented as follows:

$$r : a_1 \wedge a_2 \wedge \cdots \wedge a_i \rightarrow a_{i+1}$$

r is the name of the clause/rule [2]. We use r to refer the rule. Rule r's conditions are $a_1 \cdots a_i$. Rule r's conclusion is a_{i+1}. It is read as "If a_1 and a_2 and $\cdots$ and a_i are established, a_{i+1} must be established too."

As each Horn clause has only one conclusion, proving a thesis is a typical downward traversal of a tree formed by chainning multiple clauses together. As such, it is easy to show that proving a thesis in propositional logic can be done effectively. Horn clause logic *per se* belongs to the class of definite logic. The term 'definite' refers to the meaning of Horn clause logic. A Horn logic program has an unique interpretation characterized by Tarski semantics [3]. The existence of a 'canonical' meaning for every clauses rendered Horn clause logic popular in earlier artificial intelligence research.

The popluarity of Horn clause logic is further strengthened by the introducition of PROLOG (PROgramming in LOGic) [Colmerauer *et al.*, 1973], a Horn clause reasoning system. PROLOG was introduced nearly at the same time as Kowalski published his seminal paper [Kowalski, 1974]. Ten years later, David Warren introduced an abstract machine [Warren, 1983], the Warren Abstract Machine (WAM). WAM boosted the efficiency of PROLOG making real-life PROLOG programming practical.

More than two decades after Kowalski's pioneering work in the field of logic programming, the framework of Horn clause logic was regarded by many as too rigid for knowledge based applications. Not only does it fail to capture some practical features in knowledge representation, like incompleteness and indefiniteness, but it also behaves

[2] In the rest of this thesis, we use "rule" and "clause" interchangeably.

[3] This is different from an indefinite logic program which usually has more then one possible readings (see later).

$$\rightarrow customer(pao)$$
$$\rightarrow customer(waiip)$$
$$\rightarrow customer(william)$$
$$\rightarrow customer(charles)$$
$$\rightarrow customer(benson)$$

Figure 2.1: Example of knowledge in a database

$$penguin(t) \rightarrow bird(t)$$
$$bird(t) \rightarrow can\_fly(t)$$
$$penguin(t) \rightarrow \neg can\_fly(t)$$
$$\rightarrow penguin(t)$$

Figure 2.2: Symbolic representation of an inheritance hierarchy with exceptions

strangely in the presence of inconsistent information. Nevertheless, inconsistency is unavoidable in knowledge. In the following subsections, we highlight three deficient aspects of Horn clauses.

2.1.1 Problem with incomplete information

Consider the program of a consumer product's vendor in Figure 2.1. Based on classical Horn clause semantics, it is impossible to infer "not $customer(dongdong)$" from the program. Every piece of derivable information must be mentioned in a Horn clause program. In other words, the vendor would need to store the rest of 23 billions non-customers on this planet in order to infer "dongdong" was not a customer. However, this type of inference is widely employed in commonsense reasoning and known as default reasoning [Reiter, 1978].

2.1.2 Problem with inconsistent information

In a knowledge based system, it is not uncommon that knowledge modelled by different domain experts conflict with each other. There are conflicts which even domain experts find difficult to resolve. This is especially true when conflicts are arised because of principal difference in beliefs, e.g. athesists versus christians, and models, e.g. optimistic economists versus pessimistic economists. The program Π in Figure 2.2 is such a typical example in which the classification of a penguin and the feature of a bird is in conflict. Our expectation of "birds can fly" is inconsistent with "penguins cannot fly" where "penguins" are regarded as "birds" by *common sense.*

In Horn clause theorem proving, we can deduce anything from the program Π

$\rightarrow$ *interest_rate_rise*
interest_rate_rise $\rightarrow$ *running_cost_increase* $\vee$ *selling_index_drop*
running_cost_increate $\rightarrow$ *close_shop*
selling_index_drop $\rightarrow$ *close_shop*

Figure 2.3: Example of knowledge involving indefinite information

(Figure 2.2). Let us consider the derivation of a thesis P : *I_am_a_FBI_agent* from Π. By the principle of Davis-Putnam procedure [Davis and Putman, 1960], it is only necessary to show that a thesis $\neg P$ is inconsistent with Π. However, Π is inherently inconsistent due to $\Pi \models can\_fly(t)$ [4] and $\Pi \models \neg can\_fly(t)$. Thus, P will follow from Π as the theorem prover can derive $\perp$ from $P \cup can\_fly(t) \cup \neg can\_fly$. In fact, it is not only *I_am_a_FBI_agent* provable from Π but also *Elvis_Presley_is_still_alive* and any other beliefs.

In very large scale knowledge bases, which may probably involve tens of thousands of clauses, it is impractical to maintain global consistency due to conflicting interests between different knowledge owners. For example, it is surely unnecessary for the Pentagon and Soviet Union Military Service to change their respective knowledge bases because of inconsistency within themselves. Failure in handling inconsistent information would lead the computer to always answer "yes" to whatever statements it is asked, e.g. the computer will answer "yes" to "Is it suitable to start the third world war ?" because two tiny inconsistent propositions : "bird can fly." and "there are birds which cannot fly." In this regard, plain Horn clause logic is unacceptable to be used directly in real life applications.

2.1.3 Problem with indefinite information

Another aspect of knowledge based applications is related to the indefiniteness of information. Consider the program in Figure 2.3. By common sense reasoning, it is clear that the conclusion "*close_shop*" should be drawn after analyzing and exhausting different cases of *interest_rate_rise*. However, Horn clause logic does not allow rules like *interest_rate_rise* $\rightarrow$ *running_cost_increase* $\vee$ *selling_index_drop*. This is unacceptable in practice.

2.2 Logic programming in non-Horn clauses

The term "non-Horn clauses" here is different from the classical sense of "non-Horn clauses". "Non-Horn clauses" is classically defined as "Horn clauses" extended to allow

[4] $\models$ is the classical entailment symbol in FOL , see Appendix A.

$\rightarrow writing$
$writing \wedge \sim use\_right\_hand \rightarrow use\_left\_hand$
$writing \wedge \sim use\_left\_hand \rightarrow use\_right\_hand$

Figure 2.4: Example of knowledge involving incomplete information

more than one conclusion in a clause. We use the term "non-Horn clauses" to include all classes of clauses except "Horn clauses". In the following subsections, various solutions to the above Horn clause logic problems are outlined.

2.2.1 Reasoning under incomplete information

Ray Reiter [Reiter, 1978] and Keith Clark [Clark, 1978] have shown that real world knowledge bases are mostly "incomplete" in the sense that they do not store things they do not need. If a knowledge base is a relation R, failure of finding a clause p in R usually means $\neg R(p)$, which is known as negation-as-finite-failure (NAF). The notion of NAF is the core concept of reasoning under incomplete information. Reiter introduced the idea of Closed World Assumption (CWA) to capture this idea. CWA states "that for any n-ary relation symbol R and any n-tuple of ground terms $\alpha_1, \ldots, \alpha_n$ one may assume $\neg R(\alpha_1, \ldots, \alpha_n)$ unless the contrary can classically proved" [Witold, 1990]. In logic programming discipline, this is done by introducing a new operator $\sim$ which is called the non-provable operator or default negation. Reconsidering the program in Figure 2.1, we may introduce a rule "$\sim customer(X) \rightarrow \neg customer(X)$" to represent "Mr. X is not a customer if it cannot be proved as a customer". Nowaday, default negation $\sim$ and explicit negation $\neg$ are used together in practical reasoning frameworks.

It is easy to see that $\sim$ is neither an ordinary operator nor a truth-functional operator. Its evaluation depends on the knowledge base as a whole and on the meta-level. Consider the program in Figure 2.4, which has two interpretations: use_left_hand and use_right_hand. Thus, the introduction of $\sim$ breaks the unique model properties of definite logic.

Incidentally, logic programs that contain only default negations are called normal logic programs; and those contain both default and explicit negations are called extended logic programs [Gelfond and Lifschitz, 1990].

2.2.2 Reasoning under inconsistent information

Logical consistency assumed in early formulations of symbolic logic has been shown to be problematic. In the presence of inconsistent information, we are opened to the choices of resolving, enduring or by-passing inconsistencies. The last action is surely

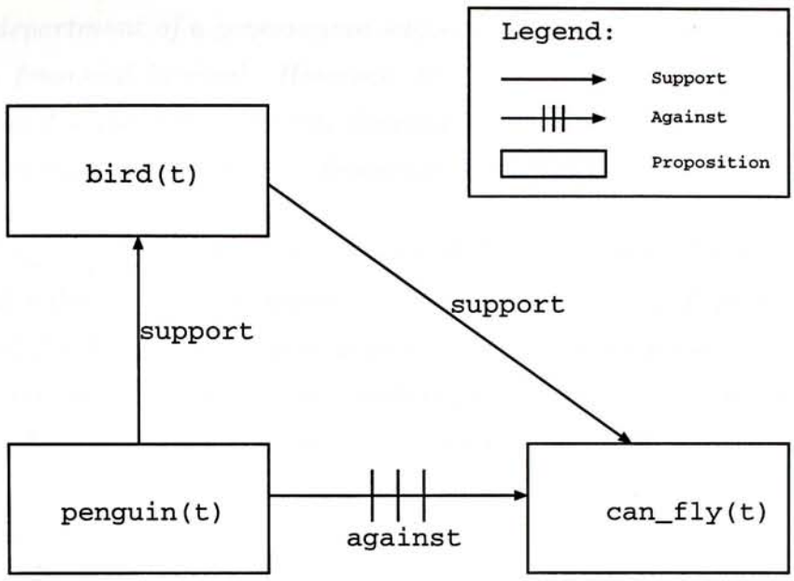

Figure 2.5: Diagram of an inheritance hierarchy with exceptions

$financial\_tumour \rightarrow financial\_policy\_decision\_required$
$financial\_policy\_decision\_required \rightarrow government\_decision$
$financial\_policy\_decision\_required \rightarrow financial\_department\_decision$
$government\_decision \rightarrow increase\_investment$
$financial\_department\_decision \rightarrow increase\_investment$

Figure 2.6: Example showing a conflict between the duty of and benefit of a government

undesirable. Currently, inconsistency resolution and endurance are under active research.

Conflicts due to inconsistencies can be resolved in several ways depending on the nature of the conflicts as well as the reasoning framework. Thomason, Horty and Touretzky proposed the ingenuious conflict resolution framework, namely the *Skeptical Inheritance Framework* [Thomason *et al.*, 1986]. Thomason et. al. proposed **specificity** as the conflict resolution metric. In a conflicting situation involving multiple sentences, the one which is most specific always wins. Reconsider the example in Figure 2.2, a schematic representation of it is shown in Figure 2.5. From Figure 2.5, it is easy to see that "bird" is less specific than "penguin" and is thus outweighted.

Specificity is a general mechanism to resolve conflicts. However, there are domains in which specificity is inappropriate. An example is the legal domain. Consider the example in Figure 2.6 which shows the following scenarios:

After some extensive cost-benefit analysis on the monetary market, the fi-

nancial department of a government suggested to stop increasing investment during a financial turmoil. However, the government, owing to its political duty and social commitment, decided to continue its effort to increase investment even at the price of financial losses.

In classical specificity analysis, the financial department's decision would override the government's decision as the former is derived from a special-purpose department which is part of the latter and thus it is more specific to the problem concerned. This was, however, counter-intuitive to our understanding of a government which should always have a higher priority on decision-making. The reason behind this was due to the conflict between duties and benefits. Duties are always "general" norms and "benefits" are always "specific" incidents. Owing to social commitment, we "should" and "ought to" prefer the more "general" norms instead of "specific" benefits in this circumstance. This example illustrates a wide class of "contrary-to-duty" problems in legal reasoning [Prakken and Sergot, 1997] which cannot be resolved by specificity. More general conflict resolution techniques, like arbitrary order or priority, are required for real-life applications.

Conflict resolution with arbitary priority is generally known as prioritized reasoning. There are no assumptions nor restrictions in the setting of priority. Specificity reasoning, on the contrary, relies on a priority restricted by a set of fixed rules. As that set of rules is domain-independent, specificity reasoning naturally fails to capture the context of reasoning , e.g. it cannot discriminate between the conflicts of zoology (e.g. Figure 2.2) and contrary-to-duty (e.g. Figure 2.6). There is no way for users to change the priority in a specificity reasoning framework. Prioritized reasoning usually mediates this through a binary relation R. A tuple consisting of two conflicting rules $< r_a, r_b >$ in R means rule r_a has a higher priority than rule r_b when they are in conflicts. In practice, there are many situations where conflicts are unavoidable and yet priority between conflicting rules are unavailable. For those cases, conflicts toleration is used to prevent disastrous result as the one discussed in Section 2.1.2.

Paraconsistent logic, pioneered by Jaskowski [Jaskowski, 1969], is a cluster of logic which tolerate conflicts. Fifteen years later, Newton da Costa introduced his formulated system [da Costa, 1963] to attack the same problem. After that, da Costa published some tens of formulations on paraconsistent logics and established the name 'paraconsistency' (see [Newton C. A. da Costa, 1995] for historical remarks on the term) to describe conflict tolerating reasoning. da Costa's approach was mainly an attempt to provide a rigorous calculus for paraconsistent logic. Alternatively, Graham Priest argued the necessity of paraconsistency as a consequence of existence of real inconsistency [Priest, 1979]. Priest's attempt was philosophical and contraversial. Nevertheless, paraconsistency is now widely recognized as a necessary feature for real-life knowledge-based

$$\rightarrow penguin(t)$$
$$penguin(t) \rightarrow bird(t)$$
$$penguin(t) \rightarrow wing\_degenerate(t)$$
$$bird(t) \rightarrow can\_fly(t)$$
$$wing\_degenerate(t) \rightarrow \neg can\_fly(t)$$

Figure 2.7: Paradox of penguin

applications.

A system is paraconsistent if it can draw non-trivial results from an inconsistent knowledge base. Consider Figure 2.7 which is a modified version of Figure 2.2. It gives an intuitive description of a penguin and shows that specificity cannot help in this situation as both "$wing\_degenerate(t)$" and "$bird(t)$" are equally specific in this setting. Classical techniques in paraconsistent logic is to extend the truth-valuation system of classical logic from 2-values to 3-values or more. For 3-valued logic, Kleene's "Strong" is a well known system [Kleene, 1952] whereas Nuel D. Belnap's bi-lattice logic is a famous reference of 4-valued semantics [Belnap, 1977]. In 3-valued logic, the extra truth value usually represents "unknown" status of a proposition and in 4-valued logic, in addition to "unknown", the fourth truth value denotes "inconsistent" status of a proposition.

2.2.3 Reasoning under indefinite information

Besides incomplete and inconsistent, real world knowledge is usually indefinite. For example, Figure 2.8 shows a gunman with a wounded right hand intending to kill another man. Rather than by-passing indefinite information, human beings use indefinite information to model possible rules. The second rule in the gunman example is an intuitive representation of indeterminate state of conclusions. It is also an usual practive to employ the technique of "reasoning about cases" over indefinite information. In the "gunman" example, we consider both cases of "a gunman holding a gun with his left hand" and "a gunman holding a gun with a wounded right hand" and conclude that the "gunman" can kill his enemy either way. Thus, the benefit of allowing indefinite information is two-folds: power in both representation and reasoning.

The line of reasoning under indefinite information is pioneered by the study of relations between logic and databases. The class of logics that allows indefinite information to be modelled is called disjunctive logic. A specific version of disjunctive logic, which combines two kinds of negations, is called extended disjunctive logic program (EDLP). It is more expressive than Horn clause logic.

The gain in reasoning power under indefinite knowledge is achieved at the expense

$\rightarrow right\_hand\_recovered$
$holding\_gun \rightarrow right\_hand\_hold \vee left\_hand\_hold$
$right\_hand\_hold \wedge right\_hand\_recovered \rightarrow kill$
$left\_hand\_hold \rightarrow kill$

Figure 2.8: Gunman with a wounded right hand

of increased complexity in both meaning and computation. It can be shown that systems with such reasoning power usually have more than one interpretation on the same piece of informatin. To define the meaning over different interpretations/readings, heuristics must be introduced. Traditionally, there are two heuristic approaches to the problem, namely credulous reading and skeptical reading. Credulous reading usually means that a thesis is established if it is established in at least one reading. Skeptical reading means that a thesis is established if it is established in all readings. The choice between credulous reading or skeptical reading is application dependent. For example, for mission critical applications, we would favour skeptical readings. In this research, we focus on skeptical reading of logics which gives results of a higher calibre in correctness.

For reasoning under skeptical reading, Michael Gelfond's Stable semantics [Gelfond and Lifschitz, 1988] and Kenneth Ross's Well-founded semantics [Ross, 1989] are most widely used. Jack Minker and Ruiz have performed a comprehensive survey for stable and well-found semantics as well as other existing frameworks of extended disjunctive logic programs [Minker and Ruiz, 1993].

2.3 Coexistence of incomplete, inconsistent and indefinite information

The difficulty of common sense reasoning is very difficult when information is not only incomplete, inconsistent and indefinite individually. When the three forms of uncertainties coexists, direct combination of individual handling methods are futile. Consider the example in Figure 2.9. It is clear that the truth value of e relies on handling methods of all three kinds of uncertainties. The groups of rules $\{r_1, r_2, r_3, r_4\}$ and $\{r_5, r_6\}$ show intricacies between incomplete information modelling and inconsistency handling. The former entails e which is inconsistent with $\neg e$ entails by the later. The later entails b which refutes $\sim b$. It is clear that methods reviewed above cannot give answer on this scenario.

There are several attempts to cover part of the scenario. Currently, stable semantics and well-founded semantics are the most widely practiced.

$r_1 :\rightarrow a$
$r_2 : a \wedge \sim b \rightarrow c \vee d$
$r_3 : c \rightarrow e$
$r_4 : d \rightarrow e$
$r_5 :\sim f \rightarrow \neg e$
$r_6 : \neg e \rightarrow b$

Figure 2.9: Co-existence of incomplete, inconsistent and indefinite information

2.4 Stable semantics

Pioneered by Michael Gelfond and Liftschtz [Gelfond and Lifschitz, 1988], stable semantics mimics the properties of stable points in mathematics. Briefly, a stable point P of a domain D is a transformation Π such that $\Pi(D, P) \Rightarrow P$ whcih means the transformation of P is itself. The transformation operator in stable semantics is the Gelfond-Liftschtz transformation Π_{GL} which takes in a set of literals P and operates on a set of logical rules D.

> $\Pi_{GL}(D, P)$ is defined as the logical consequence of D by assuming all literals involving $\sim$[5] are true if they are not in P and false otherwise. A set of literals P is a stable model of D if and only if $P = \Pi_{GL}(D, P)$.

A proposition is established with respect to a knowledge base KB if and only if it is established in every stable model of KB. In this regard, stable semantics is skeptical.

Consider a rule R involving $\sim l$ in its conditions, e.g. $R :\sim l, l_2, \ldots, l_n \rightarrow l_{n+1}$. We may read the rule as "R is the case for most of the time". The exceptional case for R is then l is provable and true. Intuitively, the set of literals involving $\sim$ are what we regarded as assumptions in daily arguments. The sentence in the definition of $\Pi_{GL}(D, P)$, i.e. "assuming all literals involving $\sim$ as true ...", can then be interpreted as pruning all rules with assumptions denied by P. This process is similar to what we do in daily arguments in which we attack our opponents' assumptions. As such, pruning of rules with weak literals [6] always succeeds. The requirement of $P = \Pi_{GL}(D, P)$ represents the self-sufficiency of P. P prunes all "unnecessary rules". The set of remaining clauses is sufficient to derive P. In this perspective, a set of literals is a stable model if and only if it is self-sufficient and attacks all unused rules.

Following the original stable semantics, several extensions have been proposed. Przymusinski extended Gelfond and Liftschtz's formulation to disjunctive logics [Przymusinski, 1991]. Przymusinski also provided a three-valued extension of stable

[5] $\sim$ is the non-provable operator in Section 2.2.1.

[6] A literal is a weak literal if it involves the non-provability sign $\sim$. It is weak because a rule consisting of such a literal can be attacked.

$$r_1 :\sim a \rightarrow b$$
$$r_2 :\sim b \rightarrow a$$
$$r_3 :\sim c \rightarrow c$$
$$r_4 : a \rightarrow c$$

Figure 2.10: A sample program showing the difficulty of computing stable models

semantics which sheded light on paraconsistent stable semantics. Chiaki Sakama and Katsumi Inoue extended stable semantics to potentially inconsistent disjunctive logic programs [Sakama and Inoue, 1995]. Sakama's extended stable semantics in four different ways which allowed, in addition to tolerating inconsistent information, computation of the preferred stable model and, introduction of suspicious and semi stable ones. However, the framework of Przymusinski and Sakama was not applicable to conflict resolution with explicit priority. Yan Zhang and Norman Foo proposed a direct extension of stable semantics [Zhang and Foo, 1997] to overcome that. But, the approach proposed did not account for reasoning with indefinite information.

As mentioned in Section 2.2.3, computation models for EDLPs is complex. The same is true for computing stable models. Consider the EDLP in Figure 2.10 showing a KB with four rules, r_1 - r_4. It is easy to show that the only stable model is $\{a, c\}$. However, although the status of a depends on r_3 and r_4, neither of them has a as its head. Thus, there is no fully top-down procedure for computing stable semantics, not even for non disjunctive cases. It should be noted that the operator Π_{GL} is anti-monotonic. Thus, it is not even possible to approach the stable model through iteration directly. Katsumi Inoue et. al. invented a bottom-up procedure [Inoue *et al.*, 1992] for calculating such stable models. Lobo and Fernández developed a nearly top-down procedure [Fernández and Lobo, 1993]. Both approaches relied on transforming the original logic program into a new one. It is worth noting that Inoue's approach exhibits high potential parallelism due to its non-backtracking bottom-up computation characteristic.

2.5 Well-founded semantics

A. Van Gelder, Kenneth Ross and J. S. Schlipf first proposed well-founded semantics in [Gelder *et al.*, 1988]. Well-founded semantics is also based on skeptical reading like stable semantics. According to Brewka [Brewka, 1996], well-founded semantics can be regarded as an efficient approximation of stable semantics and relies on a transformation operator Π_{GL} (see Section 2.4 for definition). In Brewka's formulation of well-founded semantics, the transformation operator $\Theta(D, P)$ is a double application of Π_{GL} operator, i.e. $\Theta(D, P) = \Pi_{GL}(D, \Pi_{GL}(D, P))$. A set of literals P is a well-founded model of

$$\sim economic\_up \rightarrow economic\_down$$
$$\sim economic\_down \rightarrow economic\_up$$
$$\rightarrow \neg economic\_up$$

Figure 2.11: Example showing unintuitive results drawn by well-founded semantics

$$\sim \neg b \rightarrow b$$
$$\sim \neg a \rightarrow a$$
$$\sim a \rightarrow \neg a$$

Figure 2.12: Example showing disastrous effect of conflicts in well-founded semantics

an EDLP D if and only if $P = \Theta(D, P)$.

The operator Θ involves double application of Π_{GL}. As Π_{GL} is anti-monotonic, Θ must be monotonic. Thus, the fix-point or stable point of Θ can be approached by iteration starting from an empty set. It is also possible to show that a reverse traversal method exists leading to a top-down procedure for well-founded semantics [Alferes *et al.*, 1994]. Moreover, every logic program has a well-founded model but not all logic programs have stable models. Although well-founded semantics is computationally efficient, it has some undesirable properties. Consider the program in Figure 2.10, we can show that the well-founded model is an empty set. In general, well-founded semantics draws less conclusive results than stable semantics. Besides the weakness in conclusive power, double iteration of GL operator would also introduce abnormalities.

Consider the example in Figure 2.12. It was shown that the original formulation of well-founded semantics gave an empty model in Brewka's paper [Brewka, 1996]. However, the stable model of the program is non-empty $\{b\}$ which is closer to our intuition as the conflicts of the last two rules have nothing to do with the top one. Brewka showed that such abnormality in classical well-founded semantics could be removed by a small modification to the semantics [Brewka, 1996].

Consider another example in Figure 2.11, we can see that well-founded semantics implies that $\neg$ *economic_up* is true whereas *economic_up* and *economic_down* are unknown. This is counter intuitive as we expect that *economic_down* is true and *economic_up* is false. It is unresolvable even in Brewka's modified well-founded semantics framework. This shows that the major weakness of well-founded semantics lies on its correctness.

Besides the above modifications to well-founded semantics, various extensions have been proposed to facilitate reasoning with incomplete, inconsistent and indefinite information. Ross extended well-founded semantics to handle indefinite information [Ross, 1989]. Przymusinski added classical negations into well-founded semantics

Table 2.1: Summary of difficulties and solutions of Horn clauses

DIFFICULTIES	SOLUTIONS
•Reasoning under incomplete information	•Default and Explicit Negation
•Reasoning under inconsistent information	•Paraconsistent and Prioritized Reasoning
•Reasoning under indefinite information	•Disjunctive clauses

[Przymusinski, 1990] such that reasoning with incomplete information can be done with two kinds of negations. Brewka introduced prioritized reasoning abilities into well-founded semantics [Brewka, 1996] which allow conflict resolution in reasoning with inconsistent information. Sakama added paraconsistency to well-founded semantics in [Sakama, 1992] to allow conflict toleration in reasoning with inconsistent information.

2.6 Chapter summary

In this chapter, we reviewed the three major difficulties encountered by classical Horn-clauses logics. Table 2.1 gives a summary and the corresponding possible solutions.

As we have discussed in the introduction of this chapter, knowledge in real life applications commonly consists of incomplete, inconsistent and indefinite information. A logical framework must address all of them. Extended disjunctive logic programming has been shown to be an effective tool which can satisfy this basic requirement. The desirable framework should be based on EDLP and integrated with both paraconsistent and prioritized reasoning.

From the viewpoint of computational complexity, well-founded semantics is the best existing semantics for the desirable framework. However, the inherent abnormalities in well-founded semantics is impractical for mission critical applications. For example, in applications where quality is essential , stable semantics is the ideal choice, e.g. we would certainly like a medical expert system to draw conclusions with high quality. Thus, properties of stable semantics are desirable for our framework.

In the next chapter, we will discuss a new approach, i.e. argumentation, to the problem of reasoning with incomplete, inconsistent and indefinite information. It borrows the well-practiced conflict resolution methodology from legal domain. We will shed light on new directions to tackle the aforementioned problems.

Chapter 3

Argumentation

Argumentation is a simple and yet general approach to tackle a wide class of problems in artificial intelligence research including reasoning under incomplete and inconsistent information. For this reason, we choose argumentation as our basic framework for reasoning. In this chapter, we review the theory of argumentation from the viewpoints of philosophy and artificial intelligence. We give a brief account of existing argumentation models, highlighting their inovations and related aspects to reasoning under incomplete, inconsistent and indefinite information.

Argumentation is the process of seeking out truth through dialectic. In *Republic*, Plato described dialectic as the paramount of all studies, the methodology of study. Dialectic, the backbone theory of argumentation, has been studied by philosophers for many years. It originated from the concept of dialogues in which arguments were proposed, justified and/or defeated. Its operating model and structure are closely related to verbal conversation in legal process.

Stephen Toulmin founded modern argumentation study. In his revolutionary book [Toulmin, 1958], Toulmin argued that traditional formal logics are fallacious and failed to capture complex and diversified conflicts in real life problems, e.g. contrary-to-duty problem in Section 2.2.2. Then, he went on to proposed an informal method to solve these problems. Later, Nicholas Rescher gave a formal characterization of a reasoning methodology for argumentation, dialectics, from a radically different starting point from Toulmin's, [Rescher, 1977]. Both Toulmin's and Rescher's argumentation studies did tackle the problems of reasoning with incomplete and inconsistent information, yet from a philosophical perspective. It was John Pollock who bridged argumentation and artificial intelligence. His life-time project OSCAR [Pollock, 1995, Pollock, 1996] is an attempt to build real intelligence based on argumentation. Other major contributors include Lin and Shoham, Gerard Vreeswijk, Robert Kowalski, Ronald Loui, Phan Minh Dung, Bart Verheij and Henry Prakken.

The objective of this chapter is to outline the necessary features of an argumenta-

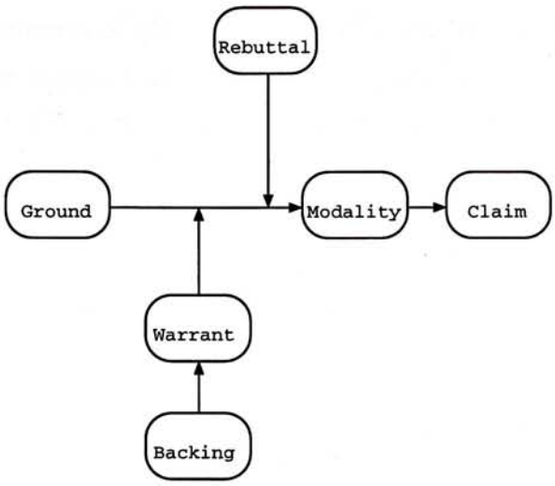

Figure 3.1: Toulmin diagram : components of an argument

tion framework which can reason over incomplete, inconsistent and indefinite information. The rest of the chapter is organized as follows: We describe the basic ideas behind Stephen Toulmin's and Rescher's (philosophical-base) argumentation frameworks in Section 3.1 and 3.2, respectively. Section 3.3 outlines the existing argumentation frameworks employed in artificial intelligence; and their pros and cons are summarised in Section 3.4.

3.1 Toulmin's informal argumentation model

Toulmin's contributions to argumentation are three-folds [Toulmin, 1958]. He identified fallacies and incapabilities of formal logic in modelling knowledge and further presented an informal setting on arguments for avoiding these problems.

Toulmin argued that *naive* formal logic "is an unrepresentative and misleadingly simple sort of arguments" [Toulmin, 1958]. As a consequence, the problems of formal logic in argumentation are caused by the "misapplication" of formal logic to commonsense reasoning. Formal logic, in Toulmin's comments, cannot cope with inconsistent information nor indefinite information which correspond to the problems we have discussed in Section 2.1.2. Toulmin then proposed his own informal theory to mediate these problems.

Toulmins' argumentation theory is based on the central idea of "Toulmin diagram", see Figure 3.1. It is a pictorial description of relations between different components of an argument namely, *warrants*, *grounds*, *claims*, *backing*, *rebuttals* and *modality*. *Claims* are inferred from *grounds* through *warrants* in argumentation which is an analogy of the way *conclusions* are inferred from *facts* through *rules* in formal logic. Different from rules in formal logic, *warrants* are defeasible and can be set aside by *rebuttals* which are

defined as counter-arguments of *claims*. *Modality* is defined as how likely a *warrant* is set aside. *Backing* is the support of *warrant*. It is easy to notice that formal logic does not have the concept of *rebuttals*, *modality* and *backing*. To defeat an argument, it is necessary to meet the *rebuttal* requirements of the argument. Such an analysis on defeat is not available in formal logic. It is interesting to note that all the six components do not assume any form of syntax. It is also applicable to informal verbal arguments.

Notice that Toulmin's work (1958) was well before the inception of logic programming (1965) and was around the time McCarthy raised the problems of commonsense reasoning [McCarthy, 1958, McCarthy and Hayes, 1969]. The formal logics [Carnap, 1950, Strawson, 1952, von Wright, 1951] attacked by Toulmin's book were mostly theory before 1953; at that time, the field of artificial intelligence was still in its very early stage. From today's point of view, Toulmin's argument theory are rudimentary yet informal [Hample, 1977]. It is impossible to characterize Toulmin's theory literally in rigorous logics. Moreover, today's formal logic includes preliminary capabilities of defeasibility analysis which are capable to tackle the problems raised by Toulmin. In the next section, we shall describe another famous argumentation model. It employs sophisticated techniques based on a new extensions of formal logic.

3.2 Rescher's formal argumentation model

In [Rescher, 1977], Nicholas Rescher attempted to give a formal characterization of argumentation. Compare to Toulmin's framework, Rescher's argumentation theory is better structured and more rigorous. Its reasoning methodology, i.e. dialectics, appears in the form of formal logic. Syntactical restrictions are explicit in the form of logical rules and propositions like formal logic.

Besides classical operators in formal logic, Rescher's framework is based on three additional operators namely, '!' *categorical assertion*, '†' *cautious assertion* and '/' *provisoed assertion*. Assertions forwarded by arguing parties form basic moves. Basic moves can further be combined into three types of complex moves. The process of argumentation is a sequence of such moves. Each move must be controversy-oriented and must be able to defeat the last move made by the opponent. Figure 3.2[1] shows the Rescher's process of argumentation. Similar to Toulmin's framework, Rescher's theory can also represent defeasible rules. Sentences modified with *categorical* and *cautious* operators represent classical logic sentences and those modified with *provisoed* operator represent defeasible ones. A *provisoed* assertion P implies that P is "generally the case" but it may be overriden by other assertions.

Besides defeasibility, Rescher analyzes four aspects of "dialectics", namely "tolerat-

[1]This diagram is an excerpt from page 19 of Rescher's book [Rescher, 1977].

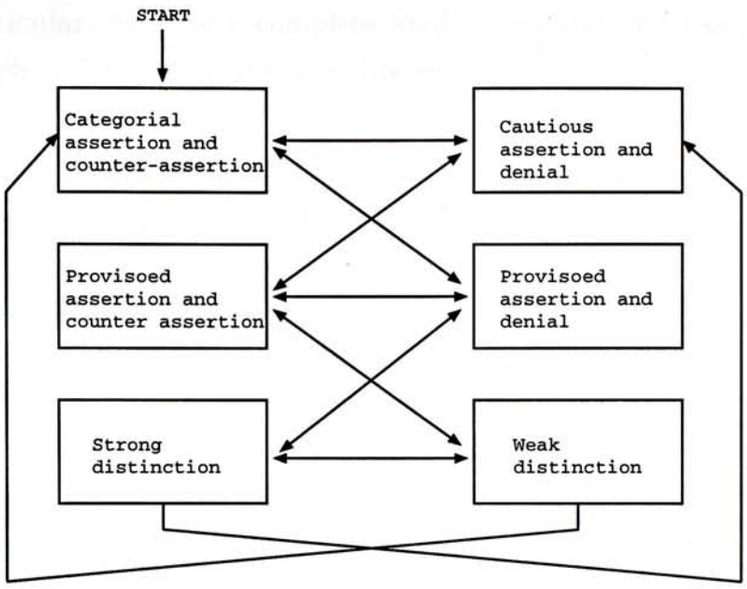

Figure 3.2: Rescher's process of argumentation
[Rescher, 1977]

Table 3.1: Summary of the four aspects of dialectics in Rescher's theory

PROBLEMS IN DIALECTICS	CORRESPONDING PROBLEMS IN LOGIC PROGRAMMING
•Tolerance of self-inconsistency	•Reasoning over inconsistent information
•Curtailing the consequences of inconsistency	•Reasoning over inconsistent information
•Potential indeterminacy	•Reasoning over indefinite information
•Constructive negation	•No comparative case

ing self-inconsistency", "curtailing the consequences of inconsistency", "potential indeterminacy" and "constructive negation". The first two concern how to tolerate inconsistency. "Potential indeterminacy" is related to the indefinite status of a proposition which can easily be obtained by considering both pieces of inconsistent information. It belongs to the class of reasoning over indefinite information. "Constructive negation" is, however, arguable. In Rescher's framework, negation is the syntactical form of rebuttal. In the process of argumentation, negation can only be applied when more supporting information is available than the previous arguments. Table 3.1 summarizes the four aspects in dialectic under Rescher's theory.

The importance of Rescher's work not only lies on the formalization of argumentation but also on its influence in artificial intelligence research. Rescher has tackled part of the problem in reasoning over inconsistent, incomplete and indefinite infor-

mation. In particular, he gave a complete analysis on the first case. Comparing to classical approach in logic programming, his analysis is still more philosophical than computational.

3.3 Argumentation in AI research

Around the same time as Rescher, several A.I. reseachers, such as John Pollock, Lin and Shoham, began to recognize that argumentation was a promising approach to commonsense reasoning. In this section, we outline several major argumentation frameworks in artificial intelligence research. In particular, we shall focus on the following aspects of existing frameworks in our discussion:

- **Reasoning over incomplete information**
  Default and Explicit Negation or other forms of incompleteness modelling techniques (see Section 2.2.1) are applicable.
- **Reasoning over inconsistent information**
  Paraconsistent and Prioritized Reasoning or other forms of conflict tolerating and resolving techniques (see Section 2.2.2) are applicable.
- **Reasoning over indefinite information**
  Disjunctive clauses or other forms of indefinite knowledge modelling techniques (see Section 2.2.3) are applicable.

We shall use the term *rebuttal* and *undercut* in the same sense as in [Prakken and Sartor, 1997].

3.3.1 Poole's *Logical Framework for Default Reasoning*

The simplest existing argumentation theory was proposed by David Poole in [Poole, 1988]. Poole's framework does not require any additional operators other than those found in propositional systems. An argument is simply a sequence of propositional rules chained together. Conclusions of an argument is the set of propositions entailed by the sequence of rules. Two arguments are in conflict if their conclusions are inconsistent with each other. Formally, a logical framework is a binary tuple $< R, H >$ where R is a consistent set of first-order rules and H a set of hypotheses in first-order logic. A theory $R \cup D$ is explainable on $< R, H >$ if D is a set of instances of H such that $R \cup D \nvdash \bot$.

Poole's framework only supports one kind of attack, namely *rebuttal.* However, its aim is not to resolve conflict by undermining a weaker one; instead it aims to separate conflicting ideas into different interpretations. In this regard, it does not support prioritized reasoning. It cannot model non-provability and thus it fails in

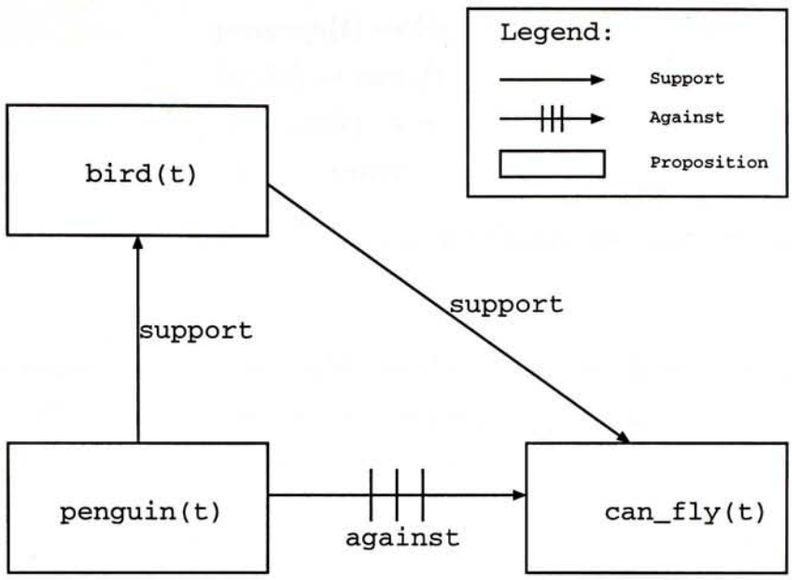

Figure 3.3: Diagram of an inheritance hierarchy with exceptions (see also Figure 3.4)

reasoning over incomplete information. It does not support reasoning over indefinite information.

3.3.2 *Inheritance Reasoning Framework* of Touretzky et. al.

Proposed by Touretzky, Horty and Thomason [Thomason *et al.*, 1986], the inheritance reasoning framework, represents the class of argumentation systems based on graph theory. Although it is not explicitly coined as an argumentation system, the intepretation of derivation paths and the conflict resolution methodology as the basic blocks of articulation are essentially argumentation-oriented. Moreover, specificity (See Section 2.1.2) is pioneered by the inheritance reasoning framework.

Touretzky et. al. shows that inheritance relation between different classes is a general knowledge model. In particular, it can easily model the derivation of a conclusion in common sense knowledge. Each path formed by chaining arcs of rules in an inheritance graph entails the derivation and the ending node on the path is the conclusion. The root node of a sub-tree formed by backward chaining from a conclusion denotes the support of the conclusion. Figure 3.3 shows a typical inheritance graph/network. Figure 3.4 shows its meaning in symbolic form. The number of arcs in a path is the metric count of the path. When conflicts present, inheritance relation and metric counts of the two conflicting paths are calculated to determine which one is more specific, e.g. $P_1 = \{penguin(t) \rightarrow bird(t), bird(t) \rightarrow can\_fly(t)\}$ is definitely less specific than $P_2 = \{penguin(t) \rightarrow \neg can\_fly(t)\}$. In this example, the conflict in $can\_fly(t)$ is resolved by P_2 overriding P_1.

$$penguin(t) \rightarrow bird(t)$$
$$bird(t) \rightarrow can\_fly(t)$$
$$penguin(t) \rightarrow \neg can\_fly(t)$$
$$\rightarrow penguin(t)$$

Figure 3.4: Symbolic representation of an inheritance hierarchy with exceptions

Inheritance reasoning does not address the notion of default negation and thus provides no means for reasoning over incomplete information. As discussed in Section 2.1.2, inheritance reasoning only provides a very restrictive mechanism, namely specificity for reasoning over inconsistent information. For reasoning over indefinite information, there is no explicit support for disjunctive logics in inheritance reasoning.

3.3.3 Pollock's *Theory of Defeasible Reasoning*

John Pollock's theory of defeasible reasoning [Pollock, 1994] is one of the most influential argumentation frameworks. Both of his argumentation-oriented approaches and the notion of an argumentation agent were novel at that time. In addition, it was Pollock who showed that bridging argumentation and artificial intelligence was a fruitful approach to common sense reasoning. Pollock described his intuition as below:

> *... philosophy has an essential role to play in artificial intelligence. The function of artificial agents is to draw conclusions and make decisions on the basis of information supplied to them. But we do not want them to draw just any old conclusions or make just any old decisions. We want them to draw rational conclusions and make rational decisions. ...* [2]

To demonstrate his idea, Pollock created the most famous argumentation system OSCAR [Pollock, 1995, Pollock, 1996] which comprised of an argumentation agent with common sense reasoning capability. OSCAR was based on his theory of defeasible reasoning. The theory borrows ideas from philosophical analysis of defeasibility. For example, it uses *prima facie* and *suppositional reasoning* to model defeasible arguments inference process of human beings. Pollock argued that human beings did not really reason with techniques in classical theorem provers. Instead, human beings were used to reason with suppositions. Suppositions represent things assumed to be true but not yet proved to be. In this way, human beings usually apply both backward chaining and forward chaining concurrently in the searching process. Classical theorem provers, on the other hand, mostly inference using either backward or forward chaining.

[2] See http://www.u.arizona.edu/~pollock/

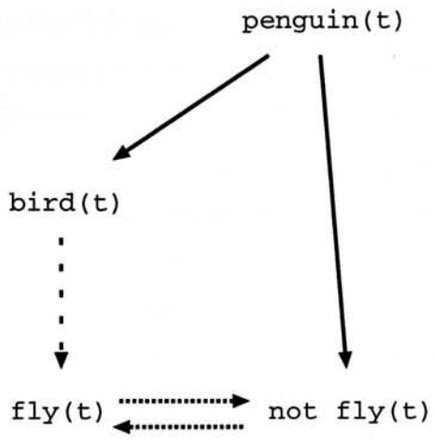

Figure 3.5: An example of inference graph

Formally, Pollock's OSCAR framework is presented in a special structure called inference graph. An inference graph is an extended inheritance network with three kinds of links, namely defeasible links - - - ->, deductive links ——> and defeat links ····>. Nodes on two ends of a link are propositions or formulas in First Order Logic (FOL). Figure 3.5 shows an inference graph depicting the example in Figure 3.4. It is not difficult to see that defeasible links and deductive links represent indefeasible and defeasible implications , respectively. A defeat link represents a source node undermining a target node. Conflicts are detected through defeat links. They are resolved according to intuitions drawn from analysis of classical paradox like lottery paradox. Semantics of Pollock's framework is based on natural deduction. He showed that OSCAR's inference was sound.

As nodes in an inference graph are FOL formulas, Pollock's framework allows reasoning over indefinite information in the form of disjunctive logic. There is no notion of "non-provable" which implies a deficiency in reasoning with incomplete information. For reasoning over inconsistent information, the concepts of *rebut* and *undercut* are supported. To resolve *rebuttal* and *undercut* conflicts, OSCAR uses probabilistic approach and forbids arbitary ordering of rules.

3.3.4 Dung's *Abstract Argumentation Framework*

Pollock's argumentation framework is rich in new concepts. Phan Minh Dung, on the contrary, attempted to provide a minimalistic argumentation framework. His argumentation theory [Dung, 1995] is a full abstract analysis of defeasibility. The framework only assumes the existence of an attacking relation between arguments and there is no syntactical restrictions on the representation language. In this way, Dung has neatly detached the subtleties of any particular logics from the framework and introduced a

generic conflict resolution methodology.

Dung's argumentation semantics and proof theory are based on the notion of *attacks* relation. An argumentation framework AF is a binary-tuple $< AR, attacks >$ where AR and *attacks* are language specific representation of arguments and the attack relation over AR respectively. Argument Arg_1 is said to attack argument Arg_2 if the tuple (Arg_1, Arg_2) is in *attacks*. The *attacks* relation models an uni-directional conflicts between two arguments. A mutual conflict occurs when two tuples differ in order, i.e. (Arg_1, Arg_2) and (Arg_2, Arg_1) model the two mutual conflicting arguments Arg_1 and Arg_2. An argument is acceptable to an argument set $ArgSet$ if and only if all of its attackers are attacked by the arguments in the argument set $ArgSet$. It becomes clear that Dung's intuition is to let every attack succeed "locally". In other words, an external order, as an arbitrator between conflicting parties of an attack, is not required. The "global" status is determined by who is the last attacker. Let Γ be an operator for mapping a set of arguments P to another set of arguments P' such that P' is acceptable to P. Γ is monotone. $Knaster - Tarski$ theorem stated that: "Suppose $(L, \leq)$ is a complete lattice, and f is any monotone map from L to L. Then f has a least-fixed point and a greatest fixed-point."[3] Therefore , a fix-point semantics for AF is available for finite AF. Further, a sound and complete proof theory for this semantics can be obtained easily by reverting the fix-point operator.

With the above minimalistic framework, Dung showed that a wide class of nonmonotonic formalisms could be subsumed through specialization of the underlying language definition and the definition of *attacks* to reflect the forms of inconsistencies. The list of subsumed formalisms included the well-known Stable Argumentation Framework, Well-founded Argumentation Framework [4] , Reiter's Default Logic and Pollock's Inductive Defeasible Logic. In [Dung and Son, 1995, Dung and Son, 1996], it was further shown that acceptability can be extended to provide prioritized reasoning mechanism, similar to Touretzky's inheritance framework in Section 3.3.2.

The major strength of Dung's framework is on its abstraction over particular logics. As such, abstract argumentation *per se* sheds no light on reasoning over indefinite information which is essentially specific to language representation. The kind of prioritized reasoning proposed in [Dung and Son, 1996] is still based on specificity which is shown to be too restrictive in Section 2.2.2. Dung does not support prioritized reasoning based on external order [Dung, 1995]. As a result, it can only partially support reasoning with inconsistent information. The abstract framework itself does not presuppose the notion of "non-provable" and thus cannot reason with incomplete information.

[3]For details of fix-point theory, readers can refer to [Piotr Rudnicki, 1996]

[4]See Chapter 2 for details.

3.3.5 Lin and Shoham's *Argument System*

Lin and Shoham's argument system (LS) [Lin and Shoham, 1989] was an attempt to subsume different nonmonotonic reasoning frameworks within argumentation theory. It was shown [Lin and Shoham, 1989] that Reiter's default logic [Reiter, 1980], Konolige's autoepistemic logic [Konolige, 1989], Clark's negation as failure [Clark, 1978] and McCarthy's circumscription [McCarthy, 1980] could all be subsumed within this Argument System.

In LS, there are two kinds of rules, namely monotonic and nonmonotonic rules. A rule takes the form of "$A_1, \ldots, A_n \rightarrow B$" or "$A$" (also known as a fact). In the rule $A_1, \ldots, A_n \rightarrow B$, $A_1 \ldots A_n$ are known as conditions whereas B is the *conclusion*. An argument is a rooted tree formed by chaining all such rules together. An argument structure T of a set of rules R is a set of arguments such that it is monontonically closed and consistent. The notion of argument structure is the core of LS. The set of aforementioned nonmonotonic frameworks are subsumed by introducing new concepts on top of the argument structure. In this way, argument structure is nothing but the intersection of subsumed semantics.

Lin and Shoham's framework enforces argument consistency thus conflicts are avoided rather than resolved as in the other cases. It does not address the issue of prioritized reasoning in avoidance of conflicts. Thus, LS does not support conflict resolution in the strict sense. LS supports reasoning over indefinite and incomplete information in two different extensions, namely subsumption of circumscription and subsumption of default logic. However, the two extension does not combined to form a unified support of reasoning over indefinite and incomplete information.

3.3.6 Vreeswijk's *Abstract Argumentation*

Vreeswijk's framework [Vreeswijk, 1991] is radically different from Dung's abstract argumentation framework. Although Dung's argumentation framework is intended to be abstract, it was essentially geared towards logic programming. Vreeswijk, on the other hand, approached the problem from a more philosophical viewpoint. He attempted to tackle the philosophical problems found in previous argumentation frameworks including Pollock's, Dung's and Loui's mentioned in [Vreeswijk, 1997].

Formally, Vreeswijk's Abstract Argumentation framework is defined on rules of definite form $\phi_1, \phi_2, \ldots, \phi_n \supset \psi$ over a language L where elements ϕ_i and ψ are members of L. The "fake" symbol $\supset$ are actually substituted with either $\rightarrow$ or $\Rightarrow$ in actual rules. $\rightarrow$ and $\Rightarrow$ denote non-defeasible and defeasible implications, respectively. An argument is a set of rules chained together through matching elements. An argument Arg is based on a member set P of L if all unchained elements on the left hand side of

rules are in P. Two or more[5] arguments $a_1, a_2, \ldots, a_n$ are in conflict if the inconsistent symbol $\perp$ is derivable from them, i.e. $a_1, a_2, \ldots, a_n \supset \perp$. Conclusive forces of conflicting arguments are calculated through an abstract function Π which is provided by the users. Vreeswijk enforced that Π must be reflexive, transitive and closed. Conflicts are resolved by giving preference to the maximal alternative(s).

There are two points deserved emphasizing in Vreeswijk's argumentation theory. First, the concept of n-ary conflict is actually nothing new to our understand of common sense reasoning. However, it was not until Vreeswijk that it was formally captured in argumentation research. Second, Vreeswijk's conflict resolution methodology is somewhat between the fully abstracted approach [Dung, 1995] and the specificity approach [Dung and Son, 1996] – i.e. it is less restrictive than the latter yet more restricted than the former. An intermediate degree of restriction is pragmatic and guarantees finite computation.

Vreeswijk's framework provides an in-depth analysis of conflicts due to inconsistency and supports prioritized reasoning. As such, it is a very promising approach to reasoning over inconsistent information. For reasoning with incomplete information, it cannot model "non-provable" knowledge. Lastly, it does not address the issue of reasoning over indefinite information at all.

3.3.7 Kowalski and Toni's *Uniform Argumentation*

Robert Kowalski and Francesca Toni's argumentation framework [Kowalski and Toni, 1996] also followed the line of abstract argumentation. Similar to Dung, they showed that argumentation could subsume most existing nonmonotonic logics like default logics [Reiter, 1980], stable semantics of extended logic programs [Gelfond and Lifschitz, 1988], nonmonotonic modal logic [McDermott, 1982] and autoepistemic logic [Moore, 1985]. Interestingly, their formulation was based not on a inconsistency oriented symbol but the notion of "non-provability". Kowalski et. al. showed that most problems in the aforementioned nonmonotonic logics could be transformed into problems of "non-provability". This gave rise to a completely different approach with many useful properties which we will mention later.

Kowalski and Toni's framework is based on definite logic programming with two kinds of negation, namely default negation and classical negation. Rules are of the form P *if* Q *and* $\sim R$ where $\sim R$ means "R is non-provable". Explicit negation of a literal is modelled by a transformation of every rule "P *if* Q *and* $\sim R$ *and* $\sim$ *not_P*" where *not_P* is a newly introduced literal denoting "the explicitly negated P". Further, the framework facilitates transformation of prioritized reasoning to a "non-provability"

[5]It is worth noting that a conflict here may involve more than two arguments. Thus, in a more general sense, it is n-ary.

form. The transformation is done by extending the concept of rule to include unique naming. Consider the following two rules:

r1 : P if Q and $\sim$ *not_P*
r2 : not_P if R and $\sim$ *P*

A preference of rule *r1* over rule *r2* can be transformed into the following program:

not_P if R and $\sim$ *P and* $\sim$ *r2_defeated*
r2_defeated if Q and $\sim$ *not_P and* $\sim$ *r1_defeated*
P if R and $\sim$ *P and* $\sim$ *r1_defeated.*

It is easy to see that *r1_defeated* and *r2_defeated* are references of other rules. Semantically, the meanings of such references must be interpreted at a level higher than other normal literals, e.g. P, Q, R. Such interpretations are provided in meta-level reasoning [6].

For reasoning over indefinite information, their framework involves only non-disjunctive logic programs and cannot model indefinite information. As it supports both default negation and transformation of explicit negation, reasoning over incomplete information is supported. Furthermore, it facilitates prioritized reasoning through transformation. Thus, it can reason with inconsistent information.

3.3.8 John Fox's *Qualitative Argumentation*

John Fox's argumentation theory, logic of argumentation *LA*, focused on reasoning over risks qualitatively [Krause P, 1993]. In [Krause P and J, 1995], he argued that quantitative reasoning like certainty factors and probability was not universal. There are domains in which quantitative measures are difficult to obtain and even if they were obtainable, their accuracy would be very low. For example, casting a measure over a segment of real number axis usually assumes a linear relation. Intermediate values are interpreted according to interpolation or alike. But, how could we quantify "like a bit" if the extreme points "hate" and "love" are valued 0 and 1, respectively. There is simply no objective measure in domains like this. Averaged value definitely introduces errors into specific cases. In qualitative domains, a qualitative measurement is usually much more desirable than an exact one. We would certainly like the expert system to tell us "like a bit" rather than 0.345687.

LA is based on minimal logic with operators & (conjunction), $\supset$ (implication) and $\perp$ (falsum) on propositional level only. Negation of a proposition or formula a ($\neg a$) is represented by $a \supset \perp$. *LA* manipulates labelled formula of the form $arg : formula$

[6] It is impossible to cover the field of meta-level reasoning in this thesis. Interested readers see [Bowen and Kowalski, 1982, Kowalski and Kim, 1991].

where *arg* is the argument supporting *formula*. Arguments are recursively defined through λ-calculus and categorical logic[7].

The strength of *LA* lies in its affinity with categorical logic. It can determine whether a proposition is *certain, confirmed, probable, plausible, supported* or *open.* The sequence represents a decreasing sequence of confidence about a proposition. It is interesting to note that six categories are defined solely on the linguistic features of *LA* and without any quantitative measure. These categories enable conflict resolution. However, linearity of the ordering sequence implies only maximal or minimal function can be used. In either way, the priority hierarchy for conflict resolution is fixed and is independent of the context of conflicts nor the underlying content.

LA does not contain the non-provability operator. As a result, it does not allow default negation to be modelled. Thus, it does not allow reasoning over incomplete information. The notion of inconsistency is core to *LA*. However, *LA* is not paraconsistent. Although *LA* does not have the problematic schema $P\&\neg P \supset Q$, the support of $\perp$ will lead to establishment of all negated propositions. Contradictory data can cause disastrous effect. In this regard, *LA* is insufficient in reasoning over inconsistent information. It is also unclear how indefinite knowledge is modelled *LA*.

3.3.9 Thomas Gordon's *Pleading Games*

Gordon's *pleading game* (*PG*) [Gordon, 1993], is a model-theorectic semantics for argumentation. It is a normative formalization and computational model of civil pleading, founded in Robert Alexy's discourse theory [Alexy, 1989].

In *PG*, an argumentation system is a tuple $< K, E >$ in which K denotes $< L, D >$. L denotes non-defeasible generic knowledge; E case-specific evidence and D the set of defeasible rules. All L, D, E are first order sentences. An argument is a set of assumptions Δ such that $\Delta \cup E \cup L \not\models \perp$ where $\models$ denotes the classical entailment relation. An argument supports Φ if $\Delta \cup E \cup L \models \Phi$. *PG* restricts the set of statements of the form of *claim*, *argument*, *rebuttal* and *denial* whose semantics is self-explanatory. The set of permitted assertions includes *concede*, *deny*, *defend* and *declare*. The set of permitted moves in a pleading game is the union of statements and assertions.

As *PG* is based on first order sentences[8], it can represent indefinite information as well as explicit negation. As a result, it supports analysis of disjunctive logic, explicit negation and undercut attacks. However, Gordon did not define how conflict toleration can be done nor did he address the issue of reasoning under incomplete information.

[7]Discussion of λ-calculus and categorical logic are out of the scope of this thesis. Interested readers can refer to [Barendregt, 1981] and [Wybraniec-Skardowska, 1991], respectively.

[8]Here we use Gordon's notion. Effectively, a first order sentence is a rule.

3.3.10 Chris Reed's *Persuasive Dialogue*

Chris Reed's argumentation theory, *Persuasive Dialogue* (*PD*), originated from the study of dialogue planning. In particular, Reed focused on how to produce an objective plan according to beliefs and intentions using a hierarchical planner. From a logical point of view, a plan for achieving an objective was similar to a proof to establish a thesis.

The central idea of Reed's argumentation theory is *persuasiveness*, the strength of an argument, which inherently assumes a multi-agent setting for discussion. In a dialogue scenario, Reed pointed out that content order in an argument are critical to persuasiveness of an argument. The responsibility of an argumentation system is then to order argument contents so as to maximize the persuasiveness. Reed borrowed ideas from psychology and rhetoric analysis, and proposed a belief framework to tackle this problem.

PD assumes non-disjunctive arguments and thus cannot model indefinite knowledge. In addition, Reed did not mention how to model incomplete and indefinite information [Reed *et al.*, 1996]. Moreover, the issue of prioritized reasoning was not addressed.

3.3.11 Ronald Loui's *Argument Game*

Ronald Loui's argument game [Loui, 1994, Loui and Chen, 1992] was based on a backward argumentation. A proposition would be supported if there was an undefeated argument supporting it. Loui's framework [Loui, 1994] could be regarded as a logical formulation of Toulmin's model with the extension of "negotiation method".

Formally, an argument is a 3-tuple $< C_i, b_i, d(C_i) >$ where C_i is a case, b_i a basis and $d(C_i)$ a claim. A basis is a set of rules. A case is an instance of facts. A claim is a defeasible conclusion inferred from a case based on a basis. Loui's definition of defeat among arguments is through classical provability operator $\vdash$ [9] as follow:

> An argument $< C_i, b_i, d(C_i) >$ defeats another argument, $< C_j, b_j, d(C_j) >$, if and only if case $d(C_i) \cup d(C_j) \vdash \perp$ and $b_j \subset b_i$ where $\perp$ is the inconsistency symbol.

It is easy to see that the above notion of defeat assumes a lower level language in which $\vdash$ is defined. Thus, Loui's argument games are generic to any logical languages with the derivation operator $\vdash$. The constraint of $b_j \subset b_i$ ensures that inconsistency is the result of adding new things. It characterizes the non-monotonic properties of an argument game, i.e. adding new information does not yield more correct results.

[9] $\vdash$ is the proof-theoretic counterpart of $\models$. For more information on proof theory, readers can refer to [Kleene, 1967].

Another interesting feature of Loui's framework is its negotiation method. This method resolves conflicts through a matrix with conflicting parties on orthogonal sides. Each side is a list of possible alternatives. Negotiation is done through finding the best combining options.

As Loui's framework only assumes a derivation operator $\vdash$ in the underlying language, it is applicable to disjunctive logic. It can cope with defeats caused by inconsistency. But it does not support defeats due to default negation. Loui's framework supports an extensive conflict resolution method through payoff matrix analysis.

3.3.12 Verheij's *Reason-Based Logics and CumulA*

Bart Verheij identified that most existing argumentation frameworks were catered for specific applications and were not general enough. He proposed Reason-Based Logic (RBL) and CumulA to attack both lower and upper levels modelling of arguments.

Similar to first order logic and propositional logic, RBL is an alternative for the underlying logic for argumentation. Contrast to existing formalisms, RBL is more descriptive and yet more complex. For example, it involves complicated conceptual analysis of arguments and different argument predicates like Applicable, Applies, Exception, Excluded, Prevails, Reason, Outweighs, Underlies [Verheij, 1996, p-34]. With these comprehensive descriptive predicates, he showed that the system could model a diverse set of argument structures.

On top of argument structures, Verheij proposed CumulA as an upper level modelling tool. CumulA operated on the abstraction of Arguments and ArgumentSchemes. Conflicts and process of resolution were modelled with similar techniques as RBL. He showed that almost all existing argumentation frameworks , like Lin and Shoham [Lin and Shoham, 1989], Vreeswijk [Vreeswijk, 1997], Pollock [Pollock, 1994], Dung [Dung, 1995], Loui [Loui and Chen, 1992], could be subsumed. The resulting framework is capable of characterizing the afore-mentioned frameworks from five aspects: type of arguments, argument structure and defeat, individual or groupwise defeat, trigger of defeat and direction of argument.

Verheij's system is strong in modelling power but weak in the following aspects: There is no real correspondence between RBL and CumulA although they are counterparts on conceptual level. Strictly speaking, there is no operational model for RBL and CumulA working as a whole. In summary, its strength is on complete analysis of arguments and argument structure instead of integrated reasoning.

3.3.13 Prakken's *Defeasible Argumentation*

Prakken's framework [Prakken and Sartor, 1997] is essentially based on argument analysis in nonmonotonic reasoning system using extended logic programming with two

kinds of negations, namely default negation $\sim$ and explicit negation $\neg$. In addition, Prakken borrowed the ideas from Brewka's extended well-founded semantics [Brewka, 1996] to come up with an argumentation theory with defeasible priority. Priority for determining result of defeat was derived dynamically through a meta-level operator. The framework not only supports nonmonotonicity in knowledge but it can also maintain the order of knowledge.

Arguments in Prakken's framework was defined as acyclic directed graph of definite logic programs. Through explicit negation and default negation, he managed to express conflict structures which were not expressible in traditional nonmonotonic reasoning framework like default logic.

For a long time, specificity was considered as one of the most important norm for conflict resolution. Prakken pointed out that this does not apply to legal domain where preference imposed by hierarchical structure is more superior. In fact, this is also true for domains whose knowledge may contain erroneous information. In Prakken's framework, conflict resolution is done through an external preference relation. In the non-defeasible variant of Prakken's argumentation system [Prakken and Sartor, 1997], preference relation are abstracted and can be combined with probabilistic mechanisms. In the defeasible variant, the preference relation is embedded in the propositional language.

With all the above features under the same framework, Prakken proved that if the input clause set was propositional and was finite in size, both non-defeasible and defeasible variants of his framework were sound and complete [Prakken and Sartor, 1997].

Prakken's framework does not have all the merits of other frameworks like rich descriptive power of CumulA, suppositional power of Pollock's defeasible reasoning framework, n-ary conflict models in Vreeswijk's abstract argumentation and abductive analysis in Kowalski's uniform argumentation framework. His framework is, however, a simple and well-balanced integrated model which caters for the two kinds of negation, prioritized reasoning and proof procedure. His framework does not account for argumentation in disjunctive setting where indefinite knowledge is modelled. Thus, it cannot analysis or resolve conflicts arised by indefinite information.

3.3.14 Summary of existing frameworks

Among the above thirteen argumentation frameworks in artificial intelligence research, we briefly classify them into pioneer works and recent works.

Pioneer works include those by Poole, Touretzky et. al., Pollock, Dung and Lin et. al. Table 3.2 summarized the features and weaknesses of these frameworks regarding to reasoning over indefinite, inconsistent and incomplete information. From Table 3.2, we see that most pioneering frameworks only concentrated on the problem

of inconsistency. Although Lin and Shoham's argument framework can cater for all three aspects, it tackles the three problems independently in different extensions rather than in an integrated system. On the contrary, Pollock's framework is an integrated approach for reasoning over indefinite and inconsistent information. The issue of integrating reasoning over incomplete information was untouched by most the pioneer frameworks. Inconsistency, indefiniteness and inompleteness are inter-wined. Their subtle relationship renders reasoning with them separately futile.

Table 3.3 summarizes recent works on argumentation in artificial intelligence. These works can be classified into three directions, namely symbolic inferencing (Vreeswijk, Kowalski, Verheij, Prakken) , qualitative analysis (Loui, John fox) and natural language processing (Chris Reed, Thomas Gordon). Chris Reed and Thomas Gordon concentrated on the relation between argumentation and dialogue systems. Loui and John Fox focused on qualitative analysis of negotiation and risks, respectively. It is worth noting that John Fox's framework already highlighted the uncertainty reasoning which naturally points to reasoning over incomplete , indefinite and inconsistent information. Vreeswijk, Kowalski et. al., Verheij and Prakken all focused on problems in symbolic inference. Prakken's framework deserves special attentions as it is close to what we want in an ideal reasoning framework.

One of the major weaknesses of Prakken's framework is its lack of support for reasoning over indefinite information. For those frameworks with this support (e.g. Pollock, Lin and Shoham, Thomas Gordon), their abilities to deal with indefinite information stem from First Order Logic. But they did not address the intricacies of indefinite information in any details similar to what we have discussed in Section 2.4 and 2.5. Despite of that, Prakken's framework provides a sophisticated and pragmatic method to reasoning over incomplete and inconsistent information.

3.4 Chapter summary

In this chapter, we have discussed two philosophical argumentation models and thirteen existing argumentation frameworks. It was shown that several argumentation frameworks are particular useful in reasoning over incomplete, inconsistent and incomplete information. They are Dung's, Lin and Shoham's, Kowalski's, Gordon's and Prakken's frameworks. Among these, Prakken's framework is most practical and simple. It inherited the merits of Dung's dialectical proof procedure. Thus, we choose Prakken's framework [Prakken and Sartor, 1997] as our basis. In our research, we focus on how to incorporate the ability of reasoning over indefinite information to the basic framework of Prakken. We shall study the relations among indefinite information, inconsistent information and incomplete information. Moreover, we shall discuss the problems in distributed artificial intelligence applications and provide possible solutions. These lay

Table 3.2: Summary of pioneer argumentation frameworks

FRAMEWORKS	REASONING WITH INCOMPLETE INFORMATION	REASONING WITH INDEFINITE INFORMATION	REASONING WITH INCONSISTENT INFORMATION
Poole's Logical Framework for Default Reasoning	×	×	o1
Touretzky et. al.'s Inheritance Network	×	×	o2.
Pollock's Theory of Defeasible Reasoning	×	√	o3
Dung's Abstract Argumentation	×	×	o4
Lin and Shoham's Argument System	o5	o5	o6

"√" denotes good support. "o" denotes fair support. "×" denotes bad or no support.

1. Only conflict avoidance. No conflict resolution.
2. Only specificity. No general priority.
3. Only probabilistic. No general priority.
4. Only specificity. Abstract proof theory. No general priority.
5. Support through semantic correspondance.
6. Only conflict avoidance.

Table 3.3: Summary of recent argumentation frameworks

FRAMEWORKS	REASONING WITH INCOMPLETE INFORMATION	REASONING WITH INDEFINITE INFORMATION	REASONING WITH INCONSISTENT INFORMATION
Vreeswijk's Abstract Argumentation	×	×	√1
Kowalski et. al.'s Abstract Argumentation	o2	×	o3
John Fox's Qualitative Argumentation	×	×	o4
Thomas Gordon's Pleading Games	×	√	o5
Chris Reed's Persuasive Dialogue	×	×	o6
Loui's Argument Game	×	×	o7
Verheij's RBL and CumulA	o	×	o
Prakken's Defeasible Argumentation	√	×	√

"√" denotes good support. "o" denotes fair support. "×" denotes bad or no support.

1. Support through abstract prioritized reasoning.
2. Support through logic programs transformation.
3. Support through program transformation.
4. Only rebuttal.
5. Only rebuttal and undercut attacks. No prioritized reasoning.
6. Support only inconsistency type.
7. Support through pay off matrix.

down the theme of the next chapter.

Chapter 4

Disjunctive Argumentation Semantics I

In this chapter, we propose an argumentation framework, Disjunctive Argumentation Semantics I (DAS-I) [Ng *et al.*, 1998b] which is motivated by the need of a unified framework for common sense reasoning over indefinite, inconsistent and incomplete information. Inspired by the work of Dung [Dung, 1995] and Prakken [Prakken and Sartor, 1997], our framework is based on logic programming as its computation mechanism is most suitable for achieving our objective. We choose Prakken's framework as our basis because it can handle reasoning with incomplete and inconsistent information in one integrated framework. In the design of DAS-I, we extend Prakken's strict argumentation framework [Prakken and Sartor, 1997] to handle indefinite information.

Prakken's original framework is based on Extended Logic Program (ELP). Our main contribution is the broadening of this base to Extended Disjunctive Logic Program (EDLP)[1]. This enables DAS-I to reason with indefinite information besides the other two types of uncertain information in a distributed setting. In this chapter, we show that integration of indefinite information handling is non-trivial. There are subtle relations between indefiniteness and inconsistency in a distributed setting. The popularity and importance of distributed agent systems make such integration essential.

This chapter is organized as follow: Section 4.1 highlights the problems in combining reasoning with indefinite information and reasoning with inconsistent information. In Section 4.2, we describe the formal definition of arguments in the DAS-I framework. This shows how we support reasoning with incomplete and indefinite information. This is followed by discussions on conflicts within a single agent and between agents in Section 4.3 and 4.4, respectively. We then discuss the semantics of DAS-I in Section

[1]See Chapter 2 or [Gelfond and Lifschitz, 1991] for details of EDLP.

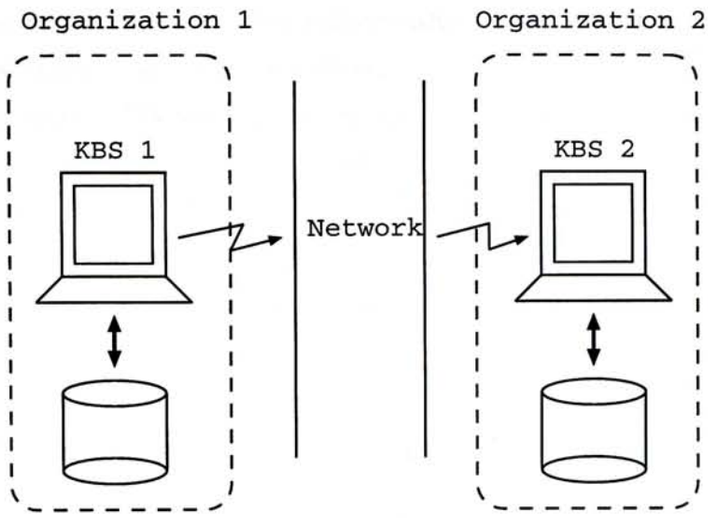

Figure 4.1: A set of distributed knowledge bases

4.5 followed by an outline of the dialectical proof theory of DAS-I in Section 4.6. The relations of DAS-I to existing frameworks are discussed in Section 4.7. In Section 4.8, we show that DAS-I is paraconsistent; and an illustrative example detailing the application of DAS-I is shown in Section 4.9. Lastly, we summarize the chapter in Section 4.10.

4.1 Background

Instead of a centralized and closed system, our argumentation framework assumes a common sense reasoning system comprising of a set of distributed knowledge based systems (KBSs) (or agents) as depicted in Figure 4.1. This setting may be due to physical distribution of knowledge/information within a single organization or due to the two KBSs being owned by two completely different organizations.

In this setting, conflicts arise in two different situations, namely

1. Intra-KBS conflicts - conflicts within one KBS; and
2. Inter-KBS conflicts - conflicts between multiple KBSs.

Intra-KBS conflicts are what we have learned in classical conflicts analysis. Conflict resolution mechanisms discussed in Chapter 2 presupposed that all knowledge reside on one KBS. Conflicts are then analyzed from a single instead of multiple view points.

Inter-KBS conflicts often exist in a network of KBSs, each managed by parties with different interests, such as different departments in a university. Knowledge in

the KBSs may overlap with each other either entirely or partially. By overlapping, we mean two or more KBSs may contain different variants of a rule. For example, "dark sky implies storm" and "dark sky implies storm or at night time" are different variants of a casual relation between "dark sky" and "storm". This overlapping usually entails different views leading to conflicts to the knowledge base. In the example of Table 4.1, the industrial development council is usually pessimistic to possible outcomes as a result of raising interest-rate whereas the banking authority council usually bears an opposite view.

Table 4.1: Different oppinions of industrial development council and banking authority council

OPINIONS OF INDUSTRIAL DEVELOPMENT COUNCIL	OPINIONS OF BANKING AUTHORITY COUNCIL
•Raising interest rate *lead to* drop on employment rate.	•Raising interest rate *lead to* increase on employment rate.
•Raising interest rate *lead to* pressure on industry.	•Raising interest rate *lead to* pressure on industry or stimulate financial related sectors.

Due to security issues and data integrity, it may not be practical to modify the KBSs of either the industrial development council or banking authority council just for avoiding such subtle conflicts. This is because other parts of the KBSs may depend on the modified knowledge. In practice, a sensible strategy in common sense reasoning in a distributed setting is to support between various views/opinions; in this way knowledge engineers can draw different conclusions out of them under different circumstances. However, discrepancies between distributed KBSs may lead to non-contradictory conflicts which is very difficult to be handled simply by classical logic.

4.2 Definition

Formally, our argumentation framework DAS-I is defined, in a top-down fashion, as follows: A literal is either an atom or an atom prepended with $\neg$. If a literal l is an atom a then $\bar{l}$ is $\neg a$. If l is $\neg a$ then $\bar{l}$ is a. A literal $\bar{l}$ is the complement or complementary counterpart of l.

An argumentation system AS is a binary-tuple $< Ags, Pg >$ where Ags is a collection of distributed argumentation KBSs/agents and Pg is a preference hierarchy between these agents. For example, an argumentation system "$AS_1 =<< Ag_1, Ag_2, Ag_3 >, << Ag_1, Ag_2 >, < Ag_2, Ag_3 >>>$" consists of three argumentation agents Ag_1, Ag_2

and Ag_3, and a preference hierarchy $<< Ag_1, Ag_2 >, < Ag_2, Ag_3 >>$.

An argumentation agent Ag is a binary-tuple $< R, P >$ where R is a set of rules and P is a preference hierarchy between these rules. For example, an argumentation agent "$Ag_1 =<< R_1, R_2 >, <>>$" consists of two rules R_1 and R_2, and an empty preference hierarchy $<>$.

A preference hierarchy P is a binary relation in which we say "s is preferred than g" (denoted by $s > g$) if and only if $< s, g > \in P$. Preference hierarchy is used when conflicts are detected between two agents or two rules of the same agent. For example, the preference hierarchy "$<< Ag_1, Ag_2 >, < Ag_2, Ag_3 >>$" of an argumentation system means agent Ag_1's conclusion is preferred than agent Ag_2's and Ag_2's is preferred than Ag_3's if they are conflicting. Consider another example, a preference hierarchy "$<< R_1, R_2 >>$" of an argumentation agent means that R_1's conclusion is preferred over R_2's if they are in conflict.

A rule is an $EDLP$ clause of the form,

$$r : a_1 \wedge \ldots \wedge a_l \wedge \sim a_{l+1} \wedge \ldots \wedge \sim a_m \rightarrow a_{m+1} \vee \ldots \vee a_n$$

where r is the name of the rule, $a_1, \ldots, a_n$ are all literals, $\sim$ is the non-provable operator [2]. Prepending $\sim$ to a literal l means the literal must be non-provable in the result for $\sim l$ to be true. Thus, $\sim$ is an assumption introducing operator. It is introduced to support common sense reasoning with indefinite information. Figure 4.2 depicts a structural view of a rule r.

It should be noted that the the term "provable" assumes an inference operation. The inference operation in classical logic is clearly defined as the classical entailment sign $\models$. In our framework, a literal l is "provable" if and only if there exists an argument with l in its conclusions. This is, however, paradoxical. In order to determine whether $\sim l$ is true, we have to determine the result first. However, we need to determine the status of $\sim l$ first before we can determine the result. Such paradox can be resolved by two levels of defeating, i.e. "defeat" and "strictly defeat" (see later).

We denote $Cd(r) = \{a_1, \ldots, a_m\}$, $Cn(r) = \{a_{m+1}, \ldots, a_n\}$, $Strong(Cd(r)) = \{a_1, \ldots, a_l\}$ and $Weak(Cd(r)) = \{a_{l+1}, \ldots, a_m\}$. $Strong(Cd(r))$ is the set of strong conditions of rule r. $Weak(Cd(r))$ is the set of weak conditions.

For example, in the rule "$r_8 : \sim criminal\_record \rightarrow \neg murderer$", $Cd(r_8)$ is { *criminal_record* }, $Cn(r_8)$ is { $\neg$*murderer* }, $Strong(Cd(r_8))$ is {} and $Weak(Cd(r_8))$ is { *criminal_record* }.

For another example, in the rule "r_5 : *finger_print* $\rightarrow$ *murderer* $\vee$ *owner*", $Cd(r_5)$

[2] $\sim a$ means a is not derivable. The non-provable operator $\sim$ is an operator commonly used in nonmonotonic logics [Clark, 1978].

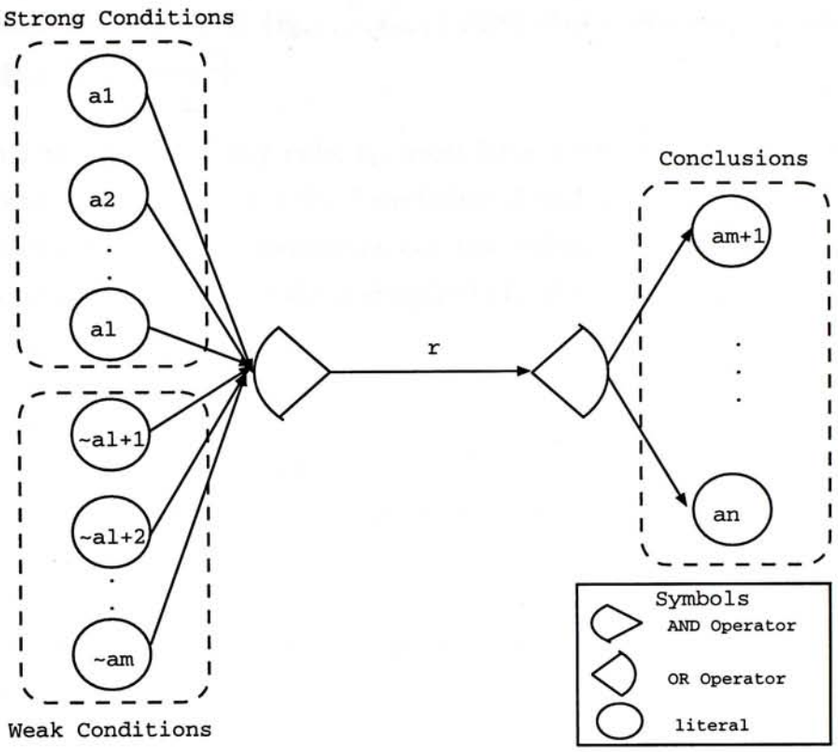

Figure 4.2: Structural view of a EDLP rule

is { *finger_print*}, $Cn(r_5)$ is { *murderer, owner* }, $Strong(Cd(r_5))$ is { *finger_print* } and $Weak(Cd(r_5))$ is {}.

We define the function $certain(r)$ under the context of an argument. It denotes a literal in the conclusion of an indefinite rule which is not pruned by the *certain*() of previous rule(s)[3]. Effectively, the function *certain* denotes a "definite" reading of an "indefinite" clause. For example, the value of $certain(r_5)$ is *murderer* in the following argument: { $r_4 : \rightarrow$ *finger_print*, r_5 : *finger_print* $\rightarrow$ *murderer* $\vee$ *owner* , r_7 : $\sim$ *ownership* $\rightarrow \neg$*owner* }; this is because "*owner*" in r_5 is pruned by r_7.

Next, we introduce the core unit for dialectical reasoning, i.e. arguments, as follows:

Definition 1 *An argument Arg is a finite sequence of rules* $\{r_0, \ldots, r_N\}$ *in which every rule* r_i *satisfies the following conditions.*

1. *If* $p \in Strong(Cd(r_i))$ *then there exists* $r \in \{r_0, \ldots, r_{i-1}\}$ *and* $p = certain(r)$.
2. *There exists* $q \in Cn(r_i)$, *denoted as* $certain(r_i)$, *such that if* $l \in \{Cn(r_i) - q\}$ *then* $\bar{l} \in \bigcup_{j=0}^{i-1} certain(r_j)$.

[3]That is, the *certain*() of the previous rule(s) is the complement of $certain(r)$, i.e. *certain(previous_rules)* = $\neg$ *certain(r)*.

3. *There does not exist* $r \in \{r_0, \ldots, r_{i-1}\}$ *such that* $certain(r_i) = certain(r)$ *and* $certain(r_i) = \overline{certain(r)}$.

Condition 1 ensures that any rule, r_i, must have been supported by previous rules or itself is grounded (i.e. when $i = 0$). Conditions 2 and 3 ensure that a rule must have a unique meaning and rules in argument are not redundant, respectively. Restriction imposed by condition 2 is for the sake of simplicity in demonstrating inter-KBS conflicts. Below is an example of an argument,

$r_0 : \rightarrow finger\_print$
$r_1 : \sim ownership \rightarrow \neg owner$
$r_2 : finger\_print \rightarrow murderer \vee owner$

To verify that it is indeed an argument, we may check the three conditions one by one as follows:

- **For condition 1** : $Strong(Cd(r_0))$ and $Strong(Cd(r_1))$ are empty and $Strong(Cd(r_2))$ is *finger_print*. $certain(r_0)$ is also *finger_print*. Thus, *finger_print* $= certain(r_0)$.
- **For condition 2** : Both $Cn(r_0)$ and $Cn(r_1)$ contain one element only. Thus only $Cn(r_2)$ needs to be considered. $certain(r_0)$ and $certain(r_1)$ are *finger_print* and $\neg owner$, respectively. Notice that $Cn(r_2)$ is $\{murderer, owner\}$ and we can conclude that $certain(r_2)$ is *murderer* as the complement of *owner* can be found in $certain(r_1)$. In summary, we have the following certain reading of r_0, r_1, r_2:
  - $certain(r_0) = finger\_print$
  - $certain(r_1) = \neg owner$
  - $certain(r_2) = murderer$
- **For condition 3** : It is obvious that there does not exist $i \neq j$ such that $certain(r_i) = certain(r_j)$ or $certain(r_i) = \overline{certain(r_j)}$.

Every argument has its own assumptions. For an argument $Arg = \{ r_0, r_1, \ldots, r_n \}$, its assumption set is the union of all weak literals of its rules, i.e. $\bigcup_{r \in Arg} Weak(Cd(r))$. Assumption set of the above argument is $\{ownership\}$. Conclusion of an argument is the set of *certain()* readings of its rules. For the above argument, its conclusion set is {*finger_print*, $\neg$*owner*, *murderer*}.

4.3 Conflicts within a KBS

Under our framework, conflicts within a KBS is referred to as intra-KBS conflicts. As an example, consider the following segment of a legal knowledge base $< R, P >$.

$R = \{$

$r_4 : \rightarrow finger\_print$
$r_5 : finger\_print \rightarrow murderer \vee owner$
$r_7 : \sim ownership \rightarrow \neg owner$
$r_8 : \sim criminal\_record \rightarrow \neg murderer$
$r_6 : murderer \rightarrow put\_into\_jail$
$r_9 : \rightarrow criminal\_record$

$\}$
$P = \emptyset$

From the above, the following arguments are found:

1. Arg_1 is comprised of r_4, r_5, and r_7.
2. Arg_2 entails Arg_1 and r_6. Irrespective of the outcome of the proof of *ownership* (r_7), Arg_2 always leads to the conclusion that the *murderer* should be *put_into_jail*. Thus, both *murderer* and *put_into_jail* are *certain*() in Arg_1.
3. Arg_3 is formed solely by r_8. Thus, *certain*() of Arg_3 (i.e. $\neg murderer$) is inconsistent with that of Arg_2 (i.e. *murderer*). This kind of head-on conflicts is called rebut.
4. Arg_4 entails r_9 alone. Note that it attacks the assumption of Arg_3. We call this undercut.

From the above example, we define the following:

Definition 2 *Arg undercuts Arg' if and only if there exists a rule r in Arg, a rule r' in Arg' and a literal p in* $Weak(Cd(r'))$ *such that* $p = certain(r)$.

Conceptually, an argument, e.g. Arg_4, attacks another, e.g. Arg_3, if and only if the former (Arg_4) rejects some assumptions of the latter (Arg_3). In the previous example, Arg_4's conclusion set is *criminal_record* whereas Arg_3's assumption set is *criminal_record*. Arg_4 supports conclusion which is assumed non-exist by Arg_3. Thus, Arg_4 is "undercutting" Arg_3.

Definition 3 *Arg rebut Arg' if and only if there exists a rule r' in Arg' and a rule r in Arg satisfying certain(r) = $\overline{certain(r')}$ and $r' > r \notin P$.*

Rebuttal attacks lie at conflicts between two arguments. An argument rebuts another if and only if *certain*() in their conclusion sets are complementary to each other. Consider the previous example again, Arg_2's conclusion set is { *finger_print*, $\neg$*owner*, *murderer*, *put_into_jail* } and Arg_3's is { $\neg$*murderer* } . Notice that the only interesting parts of the conclusions are *murderder* of Arg_2 and $\neg$*murderer* of Arg_3. The two parts are clearly inconsistent with each other. Thus, Arg_3 is rebutting Arg_2.

In general, "Arg_3 rebuts Arg_2" is not equivalent to "Arg_2 rebuts Arg_3. Rebut is essentially directional. But in the above example, Arg_3 rebuts Arg_2 also because the preference hierarchy P of $< R, P >$ is undefined/empty. In this situation, rebuts is not directional. If P is defined/non-empty, the statement "Arg_3 rebuts Arg_2" must be re-assessed under P. We must first find the set of rules involved in the conflicts, namely r_8 of Arg_3 and r_5 of Arg_2. If r_5 is not preferred over r_8, Arg_3 loses. Note that $r > r' \in P$ is not a feasible scheme. If $r > r' \in P$ replaced $r' > r \notin P$ in the above definition 3, any two arguments with complementary conclusions would not be rebuting if preference hierarchy is not present. In this case, both conflicting conclusions would be justified. Our goal is to control conflicts even when preference hierarchy is absent. Thus, justifying both conflicting conclusions is unacceptable.

Definition 4 ([Prakken and Sartor, 1997]) *Arg defeats Arg' if,*

1. *Arg is empty and, Arg' undercuts itself; or*
2. *Arg undercuts Arg'; or*
3. *Arg rebuts Arg' and Arg' does not undercut Arg.*

Case 1 of Definition 4 describes the case of a self-attacking argument Arg'. If Arg' attacks itself, it is defeated even it is the only argument in a KBS. For example, the argument $Arg = \{ r_1 : \sim a \rightarrow b, r_2 : b \rightarrow c, r_3 : c \rightarrow a \}$ must be defeated.

Case 2 is introduced to show that argument Arg_1 undercutting argument Arg_2 is sufficient to conclude a defeat even if Arg_2 attacks Arg_1 through rebuttal attacks. The justification behind this is the principle of "proof beyond any reasonable doubts". If any part of an argument's assumption set is attacked, that argument is marked "defeated" as doubts are casted by others. However, it should be noted that a "defeated" argument can be restored to "undefeated" if all of its defeaters are subsequently defeated by some undefeated arguments. In that case, the argument is said to be supported by those undefeated arguments.

Case 3 describes that rebuttal attack is insufficient to conclude a defeat. To be sufficient, an argument Arg must rebut another argument Arg' when Arg' does not

undercut Arg. This is actually the remaining situation not covered by cases 1 and 2 of Definition 4. It also shows that "undercut" is significantly stronger than "rebut" in DAS-I. To illustrates this case, let us consider the following KBS:

r_1 : $\sim$ *trading_diminishing* $\rightarrow$ *export_good*
r_2 : *export_good* $\rightarrow$ *more_employment*
r_3 : $\rightarrow$ *trading_diminishing*
r_4 : *trading_diminishing* $\rightarrow$ *business_bad*
r_5 : *business_bad* $\rightarrow$ $\neg$*more_employment*

From the above KBS, two arguments $Arg_1 = \{ r_1, r_2 \}$ and $Arg_2 = \{ r_3, r_4, r_5 \}$ are identified. By definition, Arg_2 undercuts Arg_1 and Arg_1 rebuts Arg_2. In this case, Arg_1 cannot defeat Arg_2 even Arg_1 rebuts Arg_2. The rationale is clear that Arg_2 contains a "firm" proof of *trading_diminishing* which is assumed by Arg_1 from all of its conclusions.

We then define the asymmetric order *strictly defeat* as below.

Definition 5 ([Prakken and Sartor, 1997]) *Arg strictly defeats Arg′ if and only if Arg defeats Arg′ but not Arg′ defeats Arg.*

The "strictly defeat" concept resolves the situation when argument Arg and argument Arg' defeat each other. In general, the case Arg strictly defeats Arg' and Arg' strictly defeats Arg at the same time is impossible. Thus, "strictly defeat" is asymmetric. It partially orders all arguments in a KBS into a tree form such that arguments at level n are strictly defeated by arguments at level $n-1$ as depicted in Figure 4.3.

4.4 Conflicts between KBSs

In Section 4.1, we have briefly discussed the intertwining properties of indefiniteness and inconsistency. In this section, we consider conflicts arised in an argumentation system $< Ags, Pg >$. The following two issues must be addressed in dealing with $Args$:

- **Conflicts between two different pieces of knowledge**
  This type of conflicts is classical and we have already described in the Section 4.3. These conflicts arised because rules are inconsistent, or pairs of them have complmenetary conclusions (rebut), or one rule refute the assumption of another rule (undercut). In either case, we can see that the conflicting rules are different. Either they have completely different conclusions (rebut) or they have different assumptions (undercut). Undercuts are strong enough that there are no extraneous preferences affecting its determination. Thus, we only have to concentrate on resolving rebuts which relies on preference hierarchy.

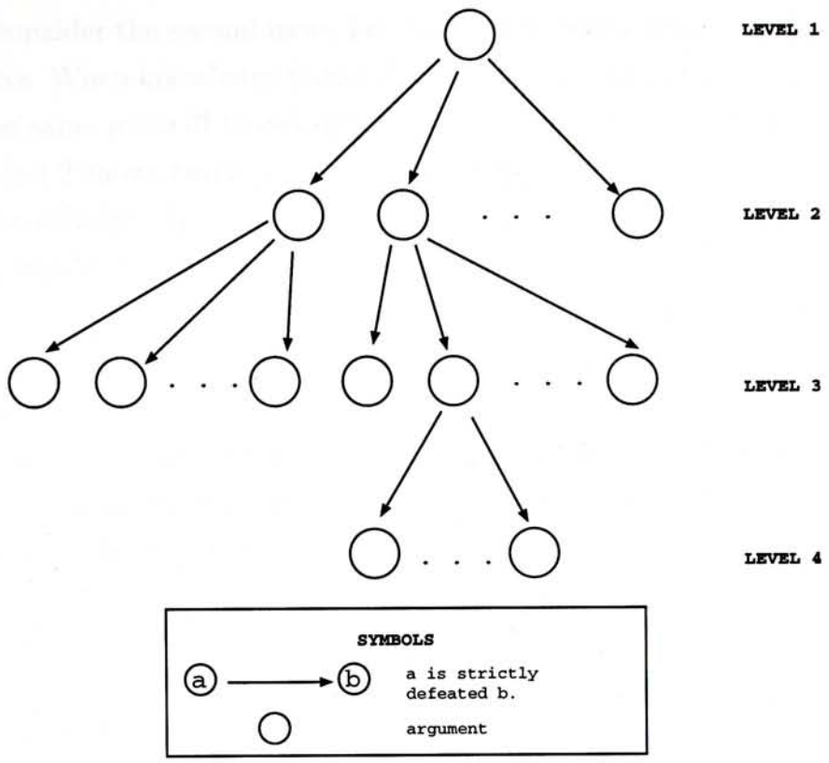

Figure 4.3: Hierarchical view of strictly defeat

- **Conflicts between two variants of the same knowledge**
  This type of conflicts is non-classical. They are arised not because of rebuts or undercuts. Rather, they are caused by the similarity of two pieces of knowledge. Consider the following scenario: Broker A told customer C that stock A would rise; and broker B told customer C that stock A or B would raise. Whom should customer C believe? In classical logic, we may represent brokers A and B as $Agt_A = \{ r_1 : \rightarrow stock\_A\_rise \}$ and $Agt_B = \{ r_2 : \rightarrow stock\_A\_rise \vee stock\_B\_rise \}$. In classical logic, r_1 undermines r_2 as r_1 is more definitive than r_2. Is this justified? If so, why?

There are many answers to the first issue, for instance probabilistic, catgorical, specificity, $\cdots$ etc. In DAS-I, we support a consistent style in conflict resolution, i.e. a preference hierarchy among KBSs is employed for resolving rebuts between different KBSs. This is where the preference hierarchy Pg in $< Ags, Pg >$ comes into play. Conflicts pertain to a single KBS can also appear in a group of KBSs. To reflect that *rebut* is based on preference hierarchy, we extend definition 3 by the following

Definition 6 *Arg of* $Ag_1 \in Ags$ *rebuts* Arg' *of* $Ag_2 \in Ags$ *if and only if there exists a rule* r_1 *in* Arg *and a rule* r_2 *in* Arg' *such that* $certain(r_1) = \overline{certain(r_2)}$ *and* $Ag_2 > Ag_1 \notin Pg$.

Next, we consider the second issue, i.e. conflicts pertaining to uncertainties between different KBSs. When knowledge is distributed over a set of agents, it is not uncommon to see that the same piece of knowledge represented differently in different agents. For example, Table 4.2 shows two agents, Agt_1 and Agt_2, each has a different interpretation of the same knowledge. It is intuitive to see that Agt_1 would conclude *put_into_jail* whereas Agt_2 would not, owing to the uncertainty in r_5. A direct merge of Agt_1 and Agt_2 would lead to *put_into_jail* as Agt_1 dominates the union in the semantics of classical logic. However, such a conclusion is based on the assumption that certain information is preferred over uncertain information. It is therefore credulous in this regard. It is arguable that on what basis can we bear this assumption and cherish certain information rather than uncertain ones? By the same token, on what basis we want to do it the other way round?

Table 4.2: Example of two distributed KBSs

AGENT	KNOWLEDGE
Agt_1	$r_1 :\rightarrow finger\_print$
	$r_2 : finger\_print \rightarrow murderer$
	$r_3 :\sim murderer \rightarrow release$
	$r_6 : murderer \rightarrow put\_into\_jail$
Agt_2	$r_4 :\rightarrow finger\_print$
	$r_5 : finger\_print \rightarrow murderer \vee owner$
	$r_6 : murderer \rightarrow put\_into\_jail$

4.4.1 Credulous View

From the viewpoint of classical logic, we can interpret a rule r as a revision of another rule r' if one shows that r' is too loose a statement (e.g. r_2 can revise r_5 in Table 4.2). We call this a credulous view of distributed knowledge.

In many traditional common sense reasoning frameworks, without the non-provability sign $\sim$, credulous view can help to enlarge the set of positive information. However, this is not the case for our framework and most other non-monotonic systems which attack default reasoning. Consider the example in Table 4.2. If we adopt the approach of classical logic, Agt_1 would out-rate Agt_2. Further, r_3 would be inapplicable and hence, *release* would not be deduced[4]. Thus, credulous view is not necessarily additive.

[4]This is deducible under skeptical view (see next section).

4.4.2 Skeptical View

When our argumentation framework is applied to domains requiring cautious answers , e.g. in legal reasoning domain, credulous inference is highly undesirable as it violates the spirit of "proof beyond any reasonable doubts". Consider the last example, if ownership is also a plausible explanation for the evidence of one's finger print on an object, then how could we arrive at the conclusion of *murderer* with certainty? In fact, it is the mission of the defense legal agent, in practice, to retrieve as many uncertain information as possible from legal knowledge bases and use them to presents "reasonable doubts" in order to attack the credibility of the accusing evidence. Under such circumstances, we would prefer Agt_2's view to Agt_1. Such preference is actually a form of suppression and reveals conflicts between different KBSs. Notice that this type of conflict arised because of the close similarity between two distributed variants of the same piece of information. In syntactic form, we introduce

Definition 7 *A rule r_1 of Agt_1 thins an argument Arg of Agt_2 at rule r_2 if and only if r_2 is in Arg and $Cd(r_2) = Cd(r_1)$ and $Cn(r_2) \subset Cn(r_1)$.*

The semantic of thinning is self-explanatory. r_1 is a more general statement about the relation between $Cd(r_2)$ and $Cn(r_2)$. It says that $Cn(r_1) - Cn(r_2)$ are possible outcomes of $Cd(r_2)$. In a modal sense, r_2 implies that given conditions of r_2 it is necessary to establish the conclusions of r_2 before r_2 is established. Thus, the necessity implied by r_2 is incorrect with respect to r_1. Semantically , r_2 in Agt_2 weakens the certainty of r_1 in Agt_1. This happens due to a skeptical view in the distributed knowledge environment. In general, we refer this as an inter-KBS conflict.

Inter-KBS-conflicts pertain to any inference schema which supports qualitative uncertainties like disjunctive in a distributed knowledge setting. A straight forward resolution scheme to inter-KBS conflicts is to eliminate the discrepant literals from the two conclusions, i.e. $Cn(r_1) - Cn(r_2)$.

For intra-KBS-conflicts, conflicting rules can easily be determined. It is, however, not the same for inter-KBS conflicts as they are arised due to the additional uncertainties (e.g. due to thinning) in the conflicting rule set. We consider a set of auxiliary rules to determine which is preferred among the conflicts rules:

1. If a rule r_1 thins an argument Arg_1 at rule r_2, the set of auxiliary rules is $auxiliary(r_1, r_2) = \{Cd(r_1) \rightarrow q | q \in (Cn(r_1) - Cn(r_2))\}$ and every rule is an alias of r_1 with the same preference hierarchy.

2. An argument Arg_1 defeats a rule r_1 if r_1 thins an argument Arg_2 at rule r_2 and for every auxiliary rule r of $auxiliary(r_1, r_2)$ there exists a rule r' in Arg_1 such that $certain(r')$ of Arg_1 is $\neg Cn(r)$.

3. An argument Arg is not defeated by a rule r_1 if r_1 thins a rule r_2 in Arg and for every auxiliary rule r in $auxiliary(r_1, r_2)$, there exists an argument defeating it.

Further, we extend "strictly defeat" in [Prakken and Sartor, 1997] as follows:

Definition 8 ([Prakken and Sartor, 1997]) *Argument Arg_1 or rule r_1 strictly defeats argument Arg_2 or rule r_2 if Arg_1 or r_1 defeats Arg_2 or r_2 but not Arg_2 or r_2 defeats Arg_1 or r_1.*

To illustrate these ideas, let us consider the example in Table 4.2 again. There is an argument $Arg_1 = \{r_1, r_2, r_6\}$ in Agt_1. By definition, r_5 and r_2 of Arg_1 give rise to inter-KBS conflict. The set of auxiliary rules is defined as,

$$
\begin{aligned}
&auxiliary(r_5, r_2) \\
&= \{\ finger\_print \rightarrow q \mid q \in (owner)\ \} \\
&= \{\ finger\_print \rightarrow owner\ \}
\end{aligned}
$$

To maintain Arg_1's undefeated status, additional arguments must be found to defeat auxiliary rules in $auxiliary(r_5, r_2)$. However, there is no such a rule in $Agt_1 \cup Agt_2$.

Similar to credulous view, skeptical view also does not give us more certain information in reasoning. Consideration of the simple example in the last subsection can help illustrate this behavior. If we consider r_5 as a thinning attacker, *put_into_jail* would not be derived from Agt_1. Thus, a skeptical view does not, in general, enlarge the set of facts concluded.

4.4.3 Generalized Skeptical View

In a skeptical view, we only focus on a particular type of similarity between disjunctive clauses. In this section, we extend the analysis to general similarity between two similar but yet distributed disjunctive clauses.

Two rules R_A and R_B are similar if and only if

1. Conditions of R_A and R_B are the same; and
2. Conclusion of R_A intersects with that of R_B.

For condition 2, we consider the following two distinguished cases:

1. *Subsumption*
   Two rules R_A and R_B are said to be similar with subsumption if and only if R_A and R_B are similar, and $Cn(R_A)$ is a proper subset of $Cn(R_B)$.

2. *Intersection*
   Two rules R_A and R_B are said to be similar with intersection if and only if R_A and R_B are similar, and $Cn(R_A) \cap Cn(R_B)$ is non-empty.

Table 4.3 shows schemas mentioned above.

Table 4.3: Conflicts analysis between similar clauses

TYPE	SCENARIO
Subsumption	$R_A : Cond \rightarrow Conc$
	$R_B : Cond \rightarrow Conc \vee \Delta$
Intersection	$R_A : Cond \rightarrow Conc \vee \Gamma$
	$R_B : Cond \rightarrow Conc \vee \Delta$

The notion of similarity is an extension of skeptical view in which subsumption is considered. Thus, we only have to analyze the case of intersection. Consider R_A and R_B in two different arguments Arg_A and Arg_B. In this extended setting, R_A and R_B are conflicting and R_B thins Arg_A. To determine the result, we consider the status of the rules involved, i.e. the unique meaning of R_A in Arg_A.

The interesting part is the *certain*() of R_A. It is possible to be in $Conc$ or Γ. In both situations, we show that a generalized skeptical view does not introduce new things and can be tackled by the following techniques:

1. Case 1 : If $certain(R_A) \in Conc$,
   - There exists arguments defeating $Cn(R_A) - certain(R_A)$.
   - $\Gamma \subseteq \{Cn(R_A) - certain(R_A)\}$.
   - Thus, there exists arguments defeating Γ.
   - R_A degenerates to $R'_A : Cond \rightarrow Conc$.
   - The problem is then reduced to the subsumption problem in the previous section
     - $R'_A : Cond \rightarrow Conc$
     - $R_B : Cond \rightarrow Conc \vee \Delta$
2. Case 2 : If $certain(R_A) \in \Gamma$,
   - There exists arguments defeating $Conc$.
   - The problem degenerates into

- $R'_A : Cond \rightarrow \Gamma$
- $R_B : Cond \rightarrow Conc \vee \Delta$

- The conflict criteria are no longer met and thus can be neglected.

The simplicity achieved in Case 2 is due to our unique meaning restriction imposed on the argument definition. The situation would be extremely complex if the restriction was relaxed.

4.5 Semantics

Now we have the bells and whistles to define our argumentation semantics for resolving conflicts under uncertainties. Formally, our semantics is based on a fix-point operator Π which operates on two sets of arguments $ArgSet$ and S. $\Pi_S(ArgSet)$ gives a subset of S such that all their counter-arguments/counter-rules are strictly defeated by arguments in Arg. It can be proved that Π is monotone in credulous view, skeptical view and generalized skeptical view. The fix-point operator Π is essentially the same as Prakken's [Prakken and Sartor, 1997]. Our semantics differ from his in the definition of "strictly defeat" which is the core concept of argumentation. Indeed, there are three kinds of fix-point operator. The key point is in what way the notion of counter-argument is interpreted. Table 4.4 summarizes the notion of counter-argument for different views shown in Sections 4.4.1 and 4.4.2.

Table 4.4: Counter-arguments for different views

	CREDULOUS	**SKEPTICAL**
COUNTER-ARGUMENTS	self-defeat	self-defeat
	undercut	undercut
	rebut	rebut
		**thinning*

To achieve monotonicity, the fix-point operator Π relies on the notion of asymmetric order (*"strictly defeat"*) which we introduced in the previous sections. By *Knaster-Tarski* theorem, the fix-point Π^* of Π with respect to a set of argument S exists and can be obtained as below.

$$F(0) = \Pi_S(\emptyset)$$
$$F(i) = \Pi_S(F(i-1))$$
$$\Pi^*_S = \lim_{i \rightarrow \infty} F(i)$$

Let *ArgSet* be the set of all arguments which can be constructed from an argumentation system AS. An argument Arg is *justified* with respect to AS if and only if $Arg \in \Pi^*_{ArgSet}$; it is *defeated* if and only if there exists a justified argument $Arg' \in AS$ which strictly defeats it; and it is *defensible*, otherwise.

4.6 Dialectical proof theory

Dung proposed a dialectical proof theory for argumentation in [Dung, 1995]. Prakken adopted it in his fixed priority framework. As the proof-theory is defined on arguments, we show that a simplified version of it can also be used in our framework.

A proof of an argument is defined on an argument tree. Each internal/external node is an argument and their child nodes are their defeaters. An argument tree T is constructed as below.

1. Level 1 is the proposition to be justified.
2. At an odd (even) level, a proponent (opponent) makes moves to strictly defeat all (any) moves from the opponents (proponents) at the previous levels.
3. In any branch, an opponent cannot make the same move twice.

A proponent is said to win a branch of an argument tree if and only if the opponent cannot make any further move at the branch; and said to win an argument tree if and only if it wins all branches. An argument Arg is *provably justified* if it wins over an argument tree at level 1; it is *provably defeated* if it is defeated by a provably justified argument; and it is *provably defensible*, otherwise. It can be shown that an argument Arg is provably justified with respect to an argumentation system AS if and only if it is justified with respect to AS [5].

4.7 Relation to existing framework

As Prakken's strict argumentation framework [Prakken and Sartor, 1997] is most similar to ours. It is easy to see that our approach degenerates to Prakken's strict argumentation framework [Prakken and Sartor, 1997] in case of only one knowledge base or a set of knowledge bases without disjunctive information. To show this, we firstly highlight the differences between our framework and Prakken's as follows:

1. **Rule**

[5] See appendix for details

- Prakken's has two kinds of implications $\rightarrow$ (strict implication) and $\Rightarrow$ (defeasible implication) whereas we only have one $\rightarrow$ (defeasible implication).
- Prakken's support definite rules and ours can handle both definite and indefinite rules.

2. **Rebuttal**

   - In Prakken, a set of rules related to a conflict is determined by traversing all strict implications only. In DAS-I, there is no such a restriction.

3. **Thinning**

   - Prakken's framework does not support the concept of distributed KBS whereas DAS-I does. Thinning attacks can only be modelled by DAS-I but not by Prakken's framework.

4. **Preference/Priority**

   - Prakken's priority is defined for single KBS only. In DAS-I, priority is defined on two levels of hierarchy, namely one for intra-KBS conflicts and another for inter-KBS conflicts.

Consider the first difference. Prakken intended to model undefeatable rules with strict implications. We argue that strict implications can be modelled by defeasible implications with priority at the top of the preference hierarchy. Consider the following example[6].

Given a KBS in Prakken's form as below:

r_1 : *dark_cloud* $\rightarrow$ *rain*
r_2 : $\Rightarrow$ *dark_cloud*
r_3 : $\Rightarrow \neg$ *rain*
Preference Hierarchy = {}

Briefly, it describes a general rule "r_1 : *dark_cloud* $\rightarrow$ *rain*" which is defeasible as there are exceptions. r_2 and r_3 are strict rules as they are facts observed at that moment. In Prakken's framework, conflicts between r_3 and r_1 are asymmetric as r_3 rebuts r_1 but not the other way round. We can re-write the above program as follows:

[6]To avoid confusion of symbols, we literally rename $\rightarrow$ as defeasible implication and $\Rightarrow$ as strict implication in the example.

r_1 : *dark_cloud* $\rightarrow$ *rain*
r_2 : $\rightarrow$ *dark_cloud*
r_3 : $\rightarrow \neg$ *rain*
Preference Hierarchy = { $r_3 > r_1$, $r_2 > r_1$ }

The preference $r_3 > r_1$ and $r_2 > r_1$ are added to reflect that r_2 and r_3 are strict rules and any defeasible rules attacking them must be defeated. In general, we can transform programs of Prakken's form into their DAS-I equivalences through this technique.

The second difference is an implication of the first difference. The DAS-I framework is an extension of Prakken's definitions and this accounts for the third difference. It is significant only when knowledge is disjunctive and is distributed to serveral different KBSs. Thus, when only one knowledge base is present, there will not be any thinning attack. In that situation, our framework reduced to an extended form of Prakken's framework, i.e. strict argumentation with disjunctive information handling. If a KBS contains only non-disjunctive rules, our framework degenerates to Prakken's with the extension of priority handling between different KBSs. When knowledge is non-disjucntive and is distributed, our framework is functionally identical to Prakken's framework without strict implication.

The fourth difference shows that our framework can cope with distributed KBSs. When only one knowledge base is available, the two-level preference hierarchy degenerates to priority relations in Prakken's framework. Further, all arguments in rebuttal conflicts are in doubt[7] if priority is unavailable. Under this situation, our framework reduces to partial semantics similar to Prakken's [Prakken and Sartor, 1997].

4.8 Issue on paraconsistency

As shown in Section 4.7, all rebuttal conflicts are arguable if priority is unavailable. In that situation, no two contradictory pair of literals can be justified. Further, we have shown in Section 4.3 that the definition of "strictly defeat" is asymmetric. Thus, no two contradictory pair of literals can be justified even if priority is present. As a result, our framework is paraconsistent.

4.9 An illustrative example

Scenario: A company would like to perform strategical planning and to seek an answer to the following question: Based on the current economic situations, is it profitable to

[7]Formally, an argument is in doubt if it is neither justified nor defeated by another justified argument. It is called a defensible argument.

start a new production line?

To proceed, the company consulted two strategic planning experts, experts A and B (see Figure 4.4.). It was hoped that individual analysis from the two independent sources could help review the intricacies of the scenario. Table 4.5 depicts the knowledge sets KB_A and KB_B extracted from experts A and B, respectively. The global order between the experts is $A > B$.

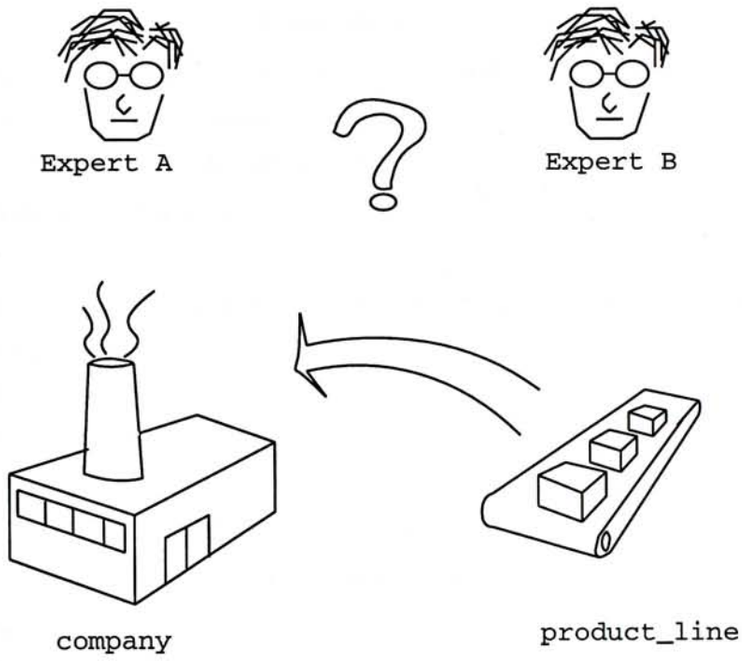

Figure 4.4: Scenario of two experts commenting on strategical actions

Analysis: In KB_A, $\{A_6, A_1, A_3, A_7, A_4, A_2\}$ form an argument Arg_1. The unique meaning of Arg_1 is *new_production_line* which suggests a new production line for the business. $\{A_7, A_5\}$ form another argument Arg_2. Arg_2 concludes that the market is stable which is contrary to the rule A_3 in Arg_1. By definition, Arg_2 rebuts Arg_1. According to KB_A's preference hierarchy, Arg_1 wins. Thus, Arg_1 is justified with respect to KB_A.

In KB_B, $\{B_9, B_1, B_4, B_2\}$ forms an argument Arg_3, $\{B_3\}$ forms Arg_4, $\{B_6, B_5\}$ forms Arg_5, and $\{B_7, B_8\}$ forms Arg_6. Arg_3 is undercut by Arg_5 which is then rebuted by Arg_6. Arg_3 undercuts Arg_4. According to the preference hierarchy, Arg_6 strictly defeats Arg_5 and Arg_3 is then justified. Arg_4 is strictly defeated by the justified argument Arg_3.

Consider the notion of skeptical view, we notice that A_1 of KB_A and B_1 of KB_B are similar knowledge about the relation among "adversary financial factor" , "economic grow" and "demand grow". By definition, B_1 thins Arg_1, the argument with A_1. The auxiliary rule is then,

Table 4.5: Knowledge bases of expert A and expert B

EXPERT	KNOWLEDGE
KB_A	$A_1 : \sim adversary\_financial\_factor \wedge economic\_grow \rightarrow demand\_grow$
	$A_2 : stable\_market \wedge demand\_grow \rightarrow new\_production\_line \vee increase\_prod\_A$
	$A_3 : \rightarrow stable\_market$
	$A_4 : \neg raw\_material\_A\_enough \rightarrow \neg increase\_prod\_A$
	$A_5 : \neg raw\_material\_A\_enough \rightarrow \neg stable\_market$
	$A_6 : \rightarrow economic\_grow$
	$A_7 : \rightarrow \neg raw\_material\_A\_enough$
	Preference Hierarchy = $\{A_3 > A_5\}$
	$B_1 : \sim adversary\_financial\_factor \wedge economic\_grow \rightarrow demand\_grow \vee competition\_grow$
	$B_2 : competition\_grow \rightarrow \neg stable\_market$
	$B_3 : \sim \neg stable\_market \wedge \sim adversary\_financial\_factor \rightarrow new\_production\_line$
	$B_4 : \sim demand\_increase \rightarrow \neg demand\_grow$
	$B_5 : interest\_raise \rightarrow adversary\_financial\_factor$
	$B_6 : \rightarrow interest\_raise$
	$B_7 : \rightarrow stock\_index\_raise$
	$B_8 : stack\_index\_raise \rightarrow \neg adversary\_financial\_factor$
	$B_9 : \rightarrow economic\_grow$
	Preference Hierarchy = $\{B_8 > B_5\}$

$$A_1^1 : \sim adversary\_financial\_factor \wedge economic\_grow \rightarrow competition\_grow$$

To defeat B_1 and restore Arg_1's justified status, we have to find arguments rebutting A_1^1. However, there is no such arguments. Thus, we have two different conclusions from KB_A and KB_B. Credulous view suggests a new production line whereas skeptical view does not suggest any strategic moves. The result is close to our intuitive understanding of credulous reasoning and skeptical reasoning.

4.10 Chapter summary

In this chapter, we have discussed the inherent assumption in classical logic [8] and how it affects distributed reasoning under uncertainties. We have proposed an integrated

[8] Certain information has higher preference than uncertain information.

framework DAS-I which features:

- **Capability for reasoning with indefinite information**
  It can modelled indefinite information in the form of disjunctive rules.
- **Capability for reasoning with inconsistent information**
  It can tolerate conflicts. Moreover, it can resolve conflicts through a general preference hierarchy. Furthermore, it can identify non-trivial conflicts, i.e. intra-KBS conflicts and inter-KBS conflicts, due to distributed settings and indefinite information.
- **Capability for reasoning with incomplete information**
  It has the default negation operator $\sim$ which facilitates modelling and employment of incomplete information.

Emphasis was paid on having these capabilities embedded in an integrated framework. This is because these types of uncertain information, although different, are closely inter-related in practice. Therefore treating them separatedly would be ineffective, if not infeasible.

In the next chapter, we shall point out that DAS-I is not perfect. It still suffers from a few abnormalities. For this reason, a revised disjunctive argumentation framework DAS-II.

Chapter 5

Disjunctive Argumentation Semantics II

In Chapter 4, we have proposed DAS-I. In this chapter, we show that some aspects in DAS-I are imperfect and propose an enhanced framework.

In the design of DAS-I, we explore argumentation in reasoning with indefinite, inconsistent and incomplete information. Our objective is to integrate the three in one holistic framework. We show that DAS-I can facilitate indefinite knowledge in the following three ways:

1. **Modelling**
   It enables direct modelling of indefinite rules.

2. **Inferencing**
   It enables indefinite rules to be used in the inference process through the concept of *certain()* which modelled how human beings handle indefinite knowledge.

3. **Consistency Checking**
   It enables detection of implicit conflicts between distributed argumentation agents.

Although the above facilities are useful in many reasoning applications, it does not support "reasoning about cases" which is commonly used in common sense reasoning. To illustrate this, let us consider the following example:

$r_1 : \rightarrow holding\_gun$
$r_2 : holding\_gun \rightarrow left\_hand\_holding \vee right\_hand\_holding$
$r_3 : left\_hand\_holding \rightarrow shot$
$r_4 : right\_hand\_holding \rightarrow shot$

It is easy to see that *shot* is justified. This is proved by "considering the different cases" of r_2's results, i.e. *left_hand_holding* and *right_hand_holding*. Both lead to the result of *shot*. This is a typical technique know as "reasoning about cases". However, *shot* cannot be deduced under the DAS-I framework. We propose Disjunctive Argumentation Semantics II (DAS-II), an extension of DAS-I with the aforesaid capability.

The above lays down the objective of this chapter. The rest of this chapter is organized as follows: In Section 5.1, we briefly describe the nature and importance of "reasoning about cases". We highlight the problems involved and further describe the philosophy behind our solution. Section 5.2 presents the formal definition of DAS-II. We then discuss the non-trivial conflicts between agents in Section 5.3. This is followed by the description of the fixpoint semantics of DAS-II in Section 5.4. In Section 5.5, we study the relations between DAS-II and other existing frameworks. The paraconsistency issue is covered in Section 5.6. A realistic example is given to illustrate the usefulness of our conflict analysis in Section 5.7. Finally, the conclusion is given in Section 5.8 [1]

5.1 Background

The idea of "reasoning about cases" is based on forward reasoning. In other words, we start from a set of premises $p_1, \cdots, p_n$ and apply forward chaining to see whether all chains end up at the same proposition q. If that is the case, q is said to be proved with respect to $p_1, \cdots, p_n$ as it is entailed for "all cases ". Figure 5.1 depicts such a scenario. The rule r_1 on the left represents the starting rule. r_1's conditions p_1, p_2,and p_3 serves as the starting premise. Rules in the middle represents intermediate chainings. The result of chaining is on the right hand side rule, i.e. Q.

In DAS-II, the notion of *certain()* no longer applies. Indeed, it is *certain()* which prevents DAS-I from supporting "reasoning about cases". *certain()* denotes a definite interpretation of a rule. It ensures that an indefinite rule must find sufficient definite information to back itself up before advancing to the subsequent inference stage(s). In other words, every indefinite rule employed must be "justified" or "backed" by definite information. Effectively, the inference process is inductive and incremental as every chaining step is more focused and leads to new definite information.

In the contrary, chaining is not focused in "reasoning about cases". By following all possible cases, it proceeds the inference in the danger of not finding anything. The chaining step is distracting and usually produces nothing definite until the ending step. Consider again the example in Figure 5.1. It could possibly take hundreds of intermediate chains just to arrive at Q. But, we do not know which is the most

[1] A simplified version of this chapter has been published ,see [Ng *et al.*, 1998a].

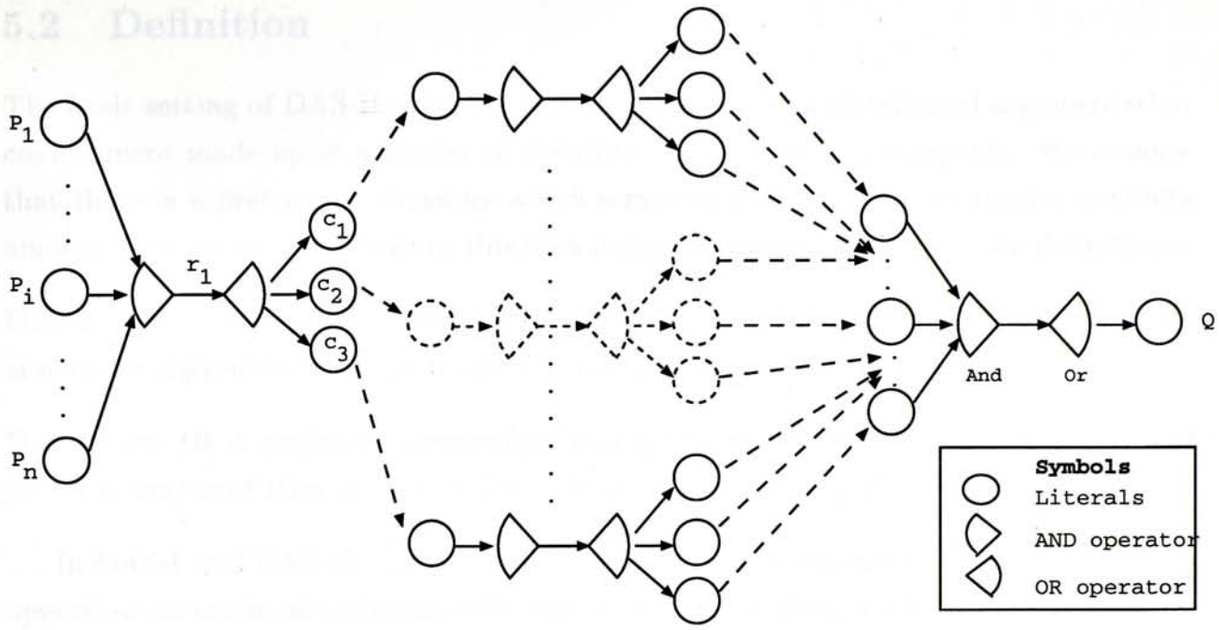

Figure 5.1: Reasoning about cases in forward chaining

effective choice point to take at every forwarding step. Forward reasoning is different from backward reasoning which is goal-oriented. The former, however, is indispensible for "reasoning about cases" in which knowledge is out of focused.

Unfocused knowledge is typical in common sense reasoning. For example, in planning, a typical AI application, indefiniteness appears at every stage involving human input. Each input may take a range of value. Proof of correctness of each planning step naturally requires the technique of "reasoning about cases".

In DAS-I, the notion of *certain* provides a definite (i.e. *certain*()) interpretation of an indefinite rule. We can, hence, mimic Prakken's strict argumentation framework on reasoning with incomplete and inconsistent information. In the design of DAS-II, the function of *certain*() is replaced by "split" [Baral and Gelfond, 1994]. Basically, "split" enumerates the possible cases of every indefinite rule in a knowledge base ($IKBS$) and produces a collection of definite knowledege bases $DKBS_1, DKBS_2, \cdots, DKBS_n$ where $DKBS_i$ is the ith split of $IKBS$. After the enumeration, one can simply concentrate on the set of definite knowledge bases instead of their indefinite counterparts. This replacement process is very useful in the construction of arguments as reasoning about definite information is easier than indefinite one. To facilitate reasoning over the DKBSs, we must revamp the definition of rebut and undercut in the light of "split".

5.2 Definition

The basic setting of DAS-II is indifferent from DAS-I, i.e. a distributed argumentation environment made up of a cluster of distributed knowledge-bases/agents. We assume that there is a preference hierarchy which serves as the arbitrator to resolve conflicts among these agents. We capture this idea formally with the following two definitions.

Definition 9 *An argumentation system AS is a binary tuple* $< Ags , Pg >$ *where Ags is a set of argumentation agents and Pg is a preference hierarchy over Ags.*

Definition 10 *A preference hierarchy Pg is a binary relation. For two objects A and B , A is preferred than B, denoted* $A > B$ *with respect to Pg if* $< A, B > \in Pg$.

In DAS-I and DAS-II, the concepts of an argumentation system are similar as the operation environments of them both involve collections of argumentation agents. Next, we define the concept of a single argumentation agent. Conceptually, an argumentation agent is an autonomous knowledge based system which consists of a knowledge base and a preference hierarchy for resolving internal conflicts. This idea is represented using the following definition.

Definition 11 *An argumentation agent Ag is a binary-tuple* $< R, P >$ *where R is a set of rules and P is a preference hierarchy over R. For rule* r_1 *and* r_2 *in R,* r_1 *is preferred over* r_2*, denoted* $r_1 >_{Agt} r_2$*, if and only if* $< r_1, r_2 > \in P$.

Table 5.1 shows an example of an argumentation agent which represents the knowledge of a financial expert. The upper part (i.e. A_1 - A_{10}) is the knowledge base and the lower part is the preference hierarchy.

5.2.1 Rules

A rule in DAS-II is an extended disjunctive logic clause. It is defined using the following operators $\vee$, $\wedge$, $\neg$, $\sim$ and $\rightarrow$. Rules represent relations between propositions/literals. A literal l is an atom a or its negation $\neg a$. Complement of a literal l, denoted as $\bar{l}$, is an atom a ($\neg a$) if l is $\neg a$ (a). Formally, a rule is defined as follows:

Definition 12 *A DAS-II rule is an extended disjunctive logic clause of the following form,* $r_1 : a_1 \wedge \cdots \wedge a_i \wedge \sim a_{i+1} \wedge \cdots \wedge \sim a_j \rightarrow a_{j+1} \vee \cdots \vee a_k$ *where* r_1 *is the name of a rule and* $a_1, \cdots a_k$ *are literals and* $\sim$ *is the non-provable operator.*

For a rule r_1 as in the above definition, $Cn(r_1) = \{ a_{j+1}, \cdots, a_k \}$, $Cd(r_1) = \{ a_1, \cdots, a_j \}$ and $Weak(Cd(r_1)) = \{ a_{i+1}, \cdots, a_j \}$ represent r_1's conclusions, conditions and weak literals, respectively. Assumptions of r_1's are also represented by the weak literals. Consider the example in Table 5.1. From the rule A_1, we obtain:

Table 5.1: A knowledge base fragment of expert A

EXPERT	KNOWLEDGE
KB_A	$A_1 : \sim adversary\_financial\_factor \wedge economic\_grow \rightarrow demand\_grow$
	$A_2 : stable\_market \wedge demand\_grow \rightarrow new\_production\_line \vee increase\_prod\_A$
	$A_3 : emmigration\_rate\_lower \rightarrow stable\_market$
	$A_4 : \neg raw\_material\_A\_enough \rightarrow \neg increase\_prod\_A$
	$A_5 : \neg raw\_material\_A\_enough \rightarrow \neg stable\_market$
	$A_6 : \rightarrow economic\_grow$
	$A_7 : \rightarrow \neg raw\_material\_A\_enough$
	$A_8 : immigation\_rate\_higher \rightarrow stable\_market$
	$A_9 : crime\_rate\_lower \rightarrow emmigration\_rate\_lower \vee immgration\_rate\_higher$
	$A_{10} : \rightarrow crime\_rate\_lower$
	Preference Hierarchy = $\{ < A_3, A_5 > \}$

$Cn(A_1) = \{ demand\_grow \}$
$Cd(A_1) = \{ adversary\_financial\_factor, economic\_grow \}$
$Weak(Cd(A_1)) = \{ adversary\_financial\_factor \}$

5.2.2 Splits

As rules may be indefinite in DAS-I, the notion of *certain()* is defined to reduce indefinite rules into definite forms but it is inapplicable for "reasoning about cases". We introduce the "split" operation in DAS-II to get around this. Before defining "split", the following definitions are required:

Definition 13 *For a set of rules s,* $CN(s) = \{ l \mid r \in s \text{ and } l \in Cn(r) \}$.

Definition 14 *For a set of rules s,* $WEAK(s) = \{ l \mid r \in s \text{ and } l \in Weak(Cd(r)) \}$.

Definition 15 *For a rule* $r : a_1 \wedge \cdots \wedge \sim a_i \rightarrow a_{i+1} \vee \cdots \vee a_k$, $Aux(r) = \{ r : a_1 \wedge \cdots \wedge \sim a_i \rightarrow a^* \mid a^* \in Cn(r) \}$.

Definition 16 *A rule r is applicable to a set of rules s if*

1. $Cd(r) \subset CN(s)$ *and*
2. $Cn(r) \cap CN(s) = \emptyset$ *and*
3. $Cn(r) \cap WEAK(s) = \emptyset$ *and*
4. $Weak(Cd(r)) \cap CN(s) = \emptyset$.

The first three restrictions define the set of conclusions $Cn()$ of all rules in s. The fourth defines that a rule r is applicable with respect to a set of rules s if:

1. it is supported, i.e. its conditions are concluded by s; and
2. it is non-redundant, i.e. none of its conclusions are concluded by s; and
3. it does not attack, i.e. none of its conclusions disproved assumptions of s; and
4. it is not attacked, i.e. non of its assumptions are disproved by s.

To illustrate the above ideas, consider the following set of rules s.

$r_1 : \rightarrow hold\_gun$
$r_2 : hold\_gun \rightarrow thief$
$r_3 : thief \rightarrow call\_police$

By definition, $CN(s) = \{ hold\_gun , thief , call\_police \}$. Consider the subset $s' = s - r_3$ of s. r_3 is applicable to s' as $CN(s') = \{ hold\_gun, thief \}$, $Cd(r_3) = \{ thief \}$ and $Cn(r_3) = \{ call\_police \}$ imply $Cd(r_3) \subset CN(s')$, $Cn(r_3) \cap CN(s') = \emptyset$ and $Weak(Cd(r)) \cap CN(s') = \emptyset$.

Definition 17 *For a rule set $R = \{r_1, \cdots, r_n\}$, we define $Split(R)$ which is a set of rule sets derived from R.*

$Split(R)$ can be computed from R using the following steps:

1. **Initial step** : $S_0(R) = \{\{\}\}$.
2. **i-th Iteration** : *If there is an applicable rule r to s, then*

$$S_{i+1}(R) = (S_i(R) - \{s\}) \cup s'$$

where s' is

$$s' = \{ s \cup r_{aux} \mid r_{aux} \in Aux(r) \} - \{ s \cup r_{aux} \mid \overline{Cn(r_{aux})} \in CN(s) \}.$$

3. **Terminate condition** : *If there is no applicable rule, then $Split(R) = S_i(R)$*

It is easy to see that "split" of a rule set is actually the least fix-point of an iterator defined in the iteration step. It consists of a set of rule sets. Each rule set must be non-disjunctive.

Proposition 1 *For a rule set R, every split of R is non-disjunctive.*

This can be shown by induction. For the base case, the only set is an empty set. At every iteration, a rule set s incorporates rules in $Aux()$, which is non-disjunctive, to form multiple rule set s'. Thus, the final fixpoint at termination must also be non-disjunctive. We shall see in the next section that this property of "split" turns out to be very useful in the analysis of conflicts between different argumentation agents.

Proposition 2 *For a rule set R, every split s of it is consistent, i.e. there does not exists l such that* $l \in CN(s)$ *and* $\overline{l} \in CN(s)$.

The notion of a split is similar to stable model semantics [Baral and Gelfond, 1994]. Next, we define the derivation symbol $\vdash$ as follows:

Definition 18 *For a rule set R,*

1. $R \vdash_S l$, *reads as "R skeptically entails S", if and only if for every split s of R,* $l \in CN(s)$; *and*
2. $R \vdash_C l$, *reads as "R credulously entails S", if and only if there exists a split s of R such that* $l \in CN(s)$.

$\vdash_S$ and $\vdash_C$ represent skeptical and credulous views on knowledge, respectively. In DAS-II, we adopt the skeptical view which can produce more sensible results. Now we have sufficient restrictions to define arguments.

Definition 19 *A rule set R is an argument if and only if*

1. *For every rule r in R, there exists a split* $s \in Split(R)$ *such that* $Aux(r) \cap s \neq \emptyset$.
2. *There exists l such that* $R \vdash_S l$.

The first condition ensures that every rules of R must participate in at least one split. For example, the rule set $\{ r_1 : left \rightarrow right , r_2 : right \rightarrow left , r_3 : \rightarrow middle \}$ cannot be an argument as r_1 and r_2 are clearly redundant. The second condition gurantees that the set of rule are conclusive.

To illustrate the concept of "split", we consider a rule set R consisting of the following rules :

$r_1 : \rightarrow hold\_gun$
$r_2 : hold\_gun \rightarrow thief \vee police$
$r_3 : thief \rightarrow go\_away$
$r_4 : police \rightarrow go\_away$
$r_5 : thief \rightarrow \neg police$

Split(R) is determined as follows:

$S_0(R) = \emptyset$

$S_1(R) = \{ \{ r_1 : \rightarrow hold\_gun \} \}$

$S_2(R) = \{$
$\{ r_1 : \rightarrow hold\_gun, r_2 : hold\_gun \rightarrow thief \}$
$\{ r_1 : \rightarrow hold\_gun, r_2 : hold\_gun \rightarrow police \}$
$\}$

$S_3(R) = \{$
$\{ r_1 : \rightarrow hold\_gun, r_2 : hold\_gun \rightarrow thief, r_3 : thief \rightarrow go\_away \}$
$\{ r_1 : \rightarrow hold\_gun, r_2 : hold\_gun \rightarrow police, r_4 : police \rightarrow go\_away \}$
$\}$

$S_4(R) = \{$
$\{ r_1 : \rightarrow hold\_gun, r_2 : hold\_gun \rightarrow thief,$
$r_3 : thief \rightarrow go\_away, r_5 : thief \rightarrow \neg police \}$
$\{ r_1 : \rightarrow hold\_gun, r_2 : hold\_gun \rightarrow police, r_4 : police \rightarrow go\_away \}$
$\}$

$S_5(R) = S_4(R)$

Split$(R) = S_5(R)$

In summary, *Split(R)* $= \{ s_1, s_2 \}$ where $s_1 = \{ r_1 : \rightarrow hold\_gun, r_2 : hold\_gun \rightarrow thief, r_3 : thief \rightarrow go\_away, r_5 : thief \rightarrow \neg police \}$ and $s_2 = \{ r_1 : \rightarrow hold\_gun, r_2 : hold\_gun \rightarrow police, r_4 : police \rightarrow go\_away \}$. It is clear that all five rules of R satisfy condition 1 of Definition 19. Also, Condition 2 of Definition 19 is satisfied as $R \vdash_S \{ hold\_gun, go\_away \}$.

5.3 Conflicts

In the previous chapter, we show that conflicts in non-disjunctive argumentation theory are only of simple type. In a distributed and disjunctive setting, conflicts are more complicated. Besides "undercut" and "rebut", there is "thinning" conflict. Figure 5.2 depicts different types of conflicts under the dimensions of "complementariness" and "balance".

The "complementariness" dimension covers the representational part of conflicts. Traditionally, conflicts means complementary ideas are derivable. This is no longer true in a distributed disjunctive agents system as we will show in the following sections. Conceptually, a conflict is complementary if and only if it involves a pair of complementary literals and it is incomplementary otherwise.

"Balance" reflects the distribution of conflicting elements. In DAS-I, every rule has one *certain*() interpretation. As conflict requires at least two parties, conflicting elements (i.e. conflicting literals) must be distributed over the two parties. Thus,

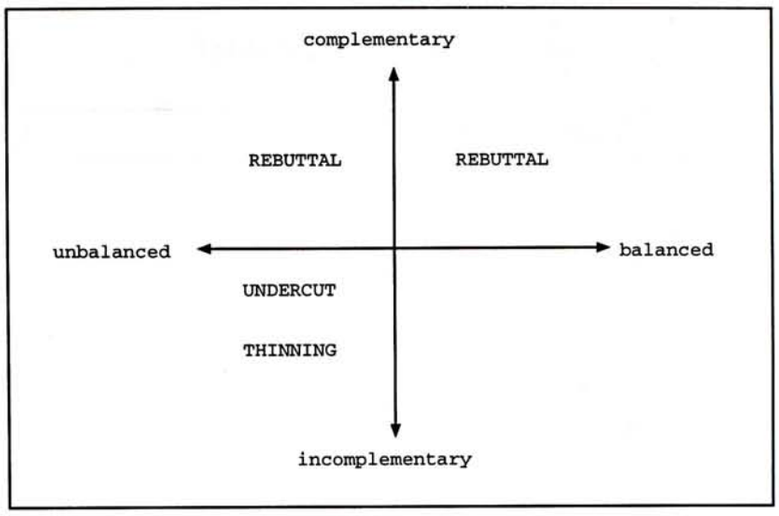

Figure 5.2: Dimensions of conflicts

conflicts in DAS-I must be balanced. Formally, a conflict is balanced if both conflicting parties contains parts of the conflicting literals, and it is unbalanced otherwise.

5.3.1 Undercut conflicts

Conceptually, an argument Arg_1 undercuts another argument Arg_2 if some conclusions originally derivable from Arg_2 are no longer derivable in the presence of Arg_1.

Definition 20 *Argument Arg_1 undercuts argument Arg_2 if and only if there exists l such that $Arg_2 \vdash_S l$ and $Arg_1 \cup Arg_2 \nvdash_S l$.*

This situation can only happen when default negation is allowed. To illustrate this, let us consider argument $Arg_A = \{A_1, A_6\}$ in Table 5.1 and argument $Arg_B = \{B_5, B_6\}$ in Table 5.3. It is clear that $Arg_A \vdash_S demand\_grow$ but $Arg_A \cup Arg_B \nvdash_S demand\_grow$. Thus, Arg_B undercuts Arg_A. From this example, it becomes apparent that undercut conflicts are incomplementary and unbalanced.

It is easy to show that our notion of argument is indeed self-sufficient and self-supporting through the following proposition. Every argument Arg undercuts its undercut conflicters formed from a subset of Arg.

Proposition 3 *For argument Arg_1, there does not exist $Arg_2 \subset Arg_1$ such that Arg_2 undercuts Arg_1.*

This can be argued as followed: As Arg_2 undercuts Arg_1, there exists l such that $Arg_1 \vdash_S l$ and $Arg_2 \cup Arg_1 \nvdash_S l$. Moreover, $Arg_2 \subset Arg_1$ implies that $Arg_2 \cup Arg_1$ is equivalent to Arg_1. Thus, $Arg_1 \nvdash_S l$.

Table 5.2: Unbalanced rebut

PROGRAM C	**PROGRAM** D
$C_1 :\rightarrow a \vee b$	$D_1 : b \rightarrow a$
$C_2 : a \rightarrow c$	$D_2 :\rightarrow b$
$C_3 : b \rightarrow \neg c$	
$C_4 : c \rightarrow d$	
$C_5 : \neg c \rightarrow d$	

As undercut conflicts are unbalanced, preference is of no concern here. We adopt the same view as Prakken [Prakken and Sartor, 1997], i.e. undercuts always suceed. In this regard, undercut has nothing to do with preference hierarchy. We define the resolution scheme of undercut conflicts as below:

Definition 21 *Argument Arg_1 defeats argument Arg_2 if Arg_1 undercuts Arg_2.*

Note that undercut conflict is the same as that in previous chapter. But rebuttal conflicts are very different from undercut.

5.3.2 Rebuttal conflicts

Rebutal conflicts are classical conflicts between two arguments. Conceptually, it arises when argument Arg_1 and argument Arg_2 are consistent by themselves but their union leads to contradiction. A contradiction is formed by a pair of complementary conclusions. Thus, rebuttal conflicts are complementary. Formally, we define it as below:

Definition 22 *An argument $Arg \vdash_S \bot$ if and only if every split s of Arg is inconsistent.*

Definition 23 *For argument Arg_1 and Arg_2, Arg_1 rebuts Arg_2 if and only if*

1. *Arg_2 does not undercut Arg_1; and*
2. *$Arg_1 \nvdash_S \bot$ and $Arg_2 \nvdash_S \bot$; and*
3. *$Arg_1 \cup Arg_2 \vdash_S \bot$.*

Although rebuttal conflicts are complementary, they are not necessary balanced. Consider the programs in Table 5.2. It is clear that the sources of contradictions in $C \cup D$, i.e. c and $\neg c$, are both arised from C instead of distributed over C and D.

To differentiate between balanced and unbalanced rebuts, the following definition is introduced.

Definition 24 *Given arguments Arg_1 and Arg_2, $owner(r, Arg_1, Arg_2)$ equals to Arg_1 or Arg_2 if r is the name of a rule in Arg_1 or Arg_2, respectively.*

$owner(r, Arg_1, Arg_2)$ represents the owner of a rule and $Relevant(Arg_1, Arg_2)$ represents the collection rule pairs with contradictory literals in each split. Upon $Relevant(Arg_1, Arg_2)$, a rebuttal conflict is determined to be balanced or unbalanced.

Definition 25 *Given arguments Arg_1 and Arg_2. If $Arg_1 \cup Arg_2 \vdash_S \bot$, the set of relevant conflicts is Relevant(Arg_1, Arg_2) = $\{ r_1, r_2 \mid r_1$ and r_2 belongs to the same split of $Arg_1 \cup Arg2$, and $Cn(r_1) = \overline{Cn(r_2)} \}$.*

Balanced rebut is the classical rebut which has been widely studied in definite argumentation framework [Prakken and Sartor, 1997]. We define it as follows:

Definition 26 *Arg_1 balancely rebuts Arg_2 if*

- *Arg_1 rebuts Arg_2.*
- *For every $< r_1, r_2 > \in Relevant(Arg_1, Arg_2)$, $r_1 \in Arg_1$ implies $r_2 \in Arg_2$, and $r_2 \in Arg_1$ implies $r_1 \in Arg_2$.*

The above definition ensures that complemenetary literals must be owned by different rules, i.e. balanced. Resolution scheme for balanced rebut involves preference hierarchy. Assume Arg_1 and Arg_2 are from agents Agt_1 and Agt_2, respectively. We define the resolution scheme formally as follows:

Definition 27 *For two arguments Arg_1 balancely rebuts Arg_2 where Arg_1 from agent Ag_1 ($=< R, P >$) and Arg_2 from Ag_2 in the same argumentation system $< Ags, Pg >$, Arg_1 defeats Arg_2*

- *if $Ag_1 = Ag_2$ and there does not exist $< r_1, r_2 > \in Relevant(\ Arg_1\ ,\ Arg_2\ )$ such that $< r_2, r_1 > \in P$ of Ag_1.*
- *if $Ag_1 \neq Ag_2$ and there does not exist $< Ag_2, Ag_1 >$ in Pg.*

To illustrate the above definition, let us consider the following arguments $Arg_1 = \{ A_5, A_7 \}$, $Arg_2 = \{A_3, A_8, A_9, A_{10}\}$ and $Arg_3 = \{ B_2, B_1, B_4, B_9\}$, see Table 5.1 and Table 5.3. It is clear that Arg_3 rebuts Arg_2 and Arg_1 rebuts Arg_2. There are two rebut relations here,

- Arg_1 balancely rebuts Arg_2. As both of them are from the same agent KB_A, the rebut is resolved according to the preference hierarchy of expert A. By definition, $Relevant(Arg_1, Arg_2) = \{ < A_5, A_3 >, < A_5, A_8 > \}$. Notice that $< A_3, A_5 > \in \tau$, Arg_1 defeats Arg_2.

- As Arg_3 is from agent B, resolution between Arg_3 and Arg_2 must be done with respect to the global preference hierarchy Pg in the argumentation system. Assuming expert A is stronger than expert B, i.e. $< Agt_1, Agt_2 > \in Pg$, Arg_2 defeates Arg_3.

In DAS-II, we identify a new kind of rebut called unbalanced rebut. Unbalanced rebut represents the difficulties of reasoning with incomplete and indefinite information. Conceptually, unbalanced rebut happened when there is a pair of derivable complementary literals embedded in one argument. Formally, we define it as follows:

Definition 28 *If argument Arg_1 rebuts argument Arg_2 and there exists $< r_1, r_2 > \in Relevant(Arg_1, Arg_2)$ such that $owner(r_1, Arg_1, Arg_2) = owner(r_2, Arg_1, Arg_2) = Arg_1$, then Arg_1 unbalancely rebuts Arg_2. Arg_1 and Arg_2 are called* **conflict bearer** *and* **conflict initiator** *, respectively.*

A conflict bearer bears a pair of complementary literals in its different splits where as a conflict initiator does not. A conflict initiator only "joins" contradictory splits with the conflict bearer. Consider the example in Table 5.2. C has two splits $s_1 = \{c \to d, a \to c, \to a\}$ and $s_2 = \{\neg c \to d, b \to \neg c, \to b\}$. Clearly, $b \to a$ of D connects s_1 and s_2.

Resolution scheme for this kind of conflicts is vague. Consider the case where a conflict initiator shows that there are potential conflicts in a conflict bearer. By skepticism, the conflict initiator always wins.

Definition 29 *If argument Arg_1 unbalancely rebuts Arg_2, Arg_1 defeats Arg_2.*

5.3.3 Thinning conflicts

Similar to undercut conflicts, thinning conflicts do not involve any complementary literal pair. Thinning conflicts are unbalanced. Conceptually, thinnning conflicts arise because of knowledge overlapping between two different agents. In multi-agent systems, overlapping of knowledge is often deliberate. Multiple perspectives on the same issue help reveal the intricacies of scenarios as well as help prevent bias caused by a single perspective. Conceptually, a thinning conflict is a relation between an argument Arg_1 in an agent Ag_1 and a rule r in another agent Ag_2. The distinction of Ag_1 and Ag_2 is significant. It is assumed that agents are the most primitive units which act on behalf of their interested parties. r thins Arg_1 if there is a rule r' in Arg_1 similar to r. The similarity mentioned here is on conceptual level. The "$\to$" sign is read as an implication symbol. Two rules are similar if they have the same set of conditions and the conclusion set of one is subsumed by the conclusion set of the other. Formally, we capture this notion as below :

Table 5.3: A knowledge base fragment of expert B

EXPERT	KNOWLEDGE
KB_B	B_1 : $\sim adversary\_financial\_factor \wedge economic\_grow \rightarrow demand\_grow \vee competition\_grow$
	B_2 : $competition\_grow \rightarrow \neg stable\_market$
	B_3 : $\sim \neg stable\_market \wedge \sim adversary\_financial\_factor \rightarrow new\_production\_line$
	B_4 : $\sim demand\_increase \rightarrow \neg demand\_grow$
	B_5 : $interest\_raise \rightarrow adversary\_financial\_factor$
	B_6 : $\rightarrow interest\_raise$
	B_7 : $\rightarrow stock\_index\_raise$
	B_8 : $stock\_index\_raise \rightarrow \neg adversary\_financial\_factor$
	B_9 : $\rightarrow economic\_grow$
	Preference Hierarchy = $\{B_8, B_5\}$

Definition 30 *For argument Arg_1 and a rule r of two different argumentation agents Ag_1 and Ag_2, respectively, r thins Arg_1 if and only if*

- $Cd(r_1) = Cd(r_2)$ *and*
- $Cn(r_2) \subset Cn(r_1)$ *and*
- *there exists l such that $Arg_1 \vdash_S l$ and $\{Arg_1 - r_2\} \cup \{r_1\} \nvdash_S l$.*

If r_1 and r_2 are similar rules, they represent different views on implication relation between conditions and conclusions. Consider Table 5.3. It represents the knowledge of an agent on behalf of expert B. B_1 is similar to A_1 in Table 5.1. Both B_1 and A_1 show the implication relation between "*demand_grow*" and "*adversary_financial_factor* $\wedge$ *economic_grow*". The difference lies at that B_1 is a more generalized statement on relations between the two. B_1 claims that "*demand_grow*" is not a necessary result of "$adversary\_financial\_factor \wedge economic\_grow$". Rather, "*competition_grow*" is also a possible outcome. Thus, B_1 thins the implication power of A_1.

As the framework adopts a skeptical view, we favour arguments which question others. Thus, thinning always succeeds.[2] Resolution scheme for thinning conflicts is then defined as below.

Definition 31 *For argument Arg_1 and a rule r, if r thins Arg_1 then Arg_1 is defeated by r.*

[2]For credulous view, the oppositie may be adopted.

The set of auxiliary rules of r and Arg_1 is constructed as follows:

Definition 32 *If a rule r_1 thins an argument Arg_1 at its rule r_2, the set of auxiliary rules is $auxiliary(r_1, Arg_1) = \{Cd(r_1) \rightarrow q | q \in (Cn(r_1) - Cn(r_2))\}$.*

Then, we define how arguments can help to restore the state of another thinned argument.

Definition 33 *An argument Arg_1 defeats a rule r_1 if r_1 thins another argument Arg_2 and for every $l \in CN(auxiliary(r_1, Arg_2))$, $Arg_2 \vdash_S \neg l$.*

5.4 Semantics

After defining three different forms of conflicts, we are now ready to work out the unified meaning of arguments of multiple distributed argumentation agents. Base on the semantics of different types of conflicts and the notion of defeat, we adopt Prakken's version of Dung's fix-point semantics as shown below.

Definition 34 ([Prakken and Sartor, 1997]) *Argument Arg_1 or rule r_1 strictly defeats argument Arg_2 or rule r_2 if Arg_1 or r_1 defeats Arg_2 or r_2 but not Arg_2 or r_2 defeats Arg_1 or r_1.*

Definition 35 ([Prakken and Sartor, 1997]) *An argument Arg is acceptable to an argument set $ArgSet$ if and only if for every argument Arg' or rule r that strictly defeats Arg there exists Arg'' in $ArgSet$ which strictly defeats Arg' or r.*

Definition 36 ([Prakken and Sartor, 1997]) *For argument sets J and $ArgSet$, $T(J, ArgSet)$ is a transformation operator which returns those arguments from $ArgSet$ that is acceptable to J. $T(J, ArgSet)$ is defined as follows:*

- $T_0(ArgSet) = T(\emptyset, ArgSet)$
- $T_{i+1}(ArgSet) = T(T_i(ArgSet), ArgSet)$
- $T^*(ArgSet) = \lim_{i \rightarrow \infty} T_i(ArgSet)$

It is easy to show that $T(J, ArgSet)$ is monotone and by Knaster-Tarski theorem, its fix-point exists and the semantics is well defined if the set of input arguments is finitary. As we assume a propositional system, there are no functions nor variables. As a result, the fix-point must exist and can be approached through iterations.

Proposition 4 ([Prakken and Sartor, 1997]) *The transformation operator $T(J, ArgSet)$ is monotone for finitary $ArgSet$.*

Table 5.4: Differences between DAS-I and DAS-II

Features	DAS-I	DAS-II
• Arguments	• Restriction : Each rule must have a *certain*() value.	• Restriction : Each arguments must at least skeptically entails one literal.
• Rebuts	• Only balanced rebuts.	• Both balanced and unbalanced rebuts.
• Proof theory	• Sound and complete proof theory.	• No proof theory.

Definition 37 ([Prakken and Sartor, 1997]) *With respect to an argument set ArgSet, an argument Arg is*

- *justified if and only if Arg is in $T^*(ArgSet)$.*
- *defeated if and only if it is strictly defeated by a justified argument Arg'.*
- *defensible if and only if it is neither justified nor defeat.*

In Section 5.7, we shall present a working example which illustrates how our framework works as a whole.

5.5 Relation to existing frameworks

To see how DAS-II relates to existing frameworks, we examine its relation to skeptical DAS-I. In the rest of thesis, we shall use DAS-I to refer skeptical view of DAS-I. Table 5.4 and Table 5.5 depict how DAS-II differs from and similar to DAS-I, respectively. From Table 5.5, it is clear that DAS-II is a close successor of DAS-I. DAS-II differs from DAS-I mostly on its definition of argument. DAS-II is more general and covers a larger set of arguments than DAS-I. This is due to the following proposition.

Proposition 5 *A* DAS-I *argument which does not undercut itself is also a* DAS-II *argument.*

Self-undercutting DAS-I arguments are defeated by definition. Therefore, their conclusions are non-provable. Moreover, they are not arguments under DAS-II by definition. Thus, their conclusions are also non-provable. For a DAS-I argument which does not undercut itself, we can inductively show that it has only one split. By definition of DAS-I, its conclusion must be consistent. As a result, it must be consistent under DAS-II. Further, the conclusions in this only one split must be skeptically entailed under

Table 5.5: Similarities between DAS-I and DAS-II

Features	DAS-I	DAS-II
• Conflicts	• Support modelling of Undercut, Rebut and Thinning conflicts.	• Suport modelling of Undercut, Rebut and Thinning conflicts.
• Resolution	• Support two level preference hierarchy.	• Support two level preference hierarchy.
• Defeats	• Prakken's Asymmetric Strictly Defeat.	• Prakken's Asymmetric Strictly Defeat.

DAS-II. As *certain*() is non-empty under DAS-I, the skeptically entailed conclusion set must also be non-empty. This completes the argument of the above proposition.

Theorem 1 *An argument justified under* DAS-I *is also justified under* DAS-II.

This theorem confirms that DAS-II is significantly more general than DAS-I. To show that this theorem is correct, let's suppose that it is not true. There must be an argument Arg_1 which is justified under DAS-I but not DAS-II. This can either be that Arg_1 is not an argument or Arg_1 is not justified under DAS-II. The former is impossible by Proposition 5. For the latter, there must be another argument Arg_2 or rules r in conflict with Arg_1. If the conflict is undercut, balancely rebut or thinning, it should have been the same as in DAS-I. This would contradict the assumption that Arg_1 is justified in DAS-I. For this reason, the conflict must be an unbalanced rebut. As Arg_1 has only one split, the pair of complementary literals must be in the same split. This also contradicts the definition of DAS-I arguments. Thus, the contrary of Theorem 1 must be false.

We have shown that DAS-I degenerated to Prakken's strict argumentation framework when all knowledge was non-disjunctive. The same is also true for DAS-II. This is because DAS-II extends DAS-I on the ground of disjunctive knowledge. Without disjunctive knowledge, DAS-II degenerates into DAS-I and further to Prakken's strict argumentation. In that situation, DAS-II holds whatever DAS-I does.

5.6 Issue on paraconsistency

As a successor of DAS-I, DAS-II is also paraconsistent. The new features and concepts introduced in DAS-II do not undermine this nice property. Formally, we have the following proposition.

Proposition 6 DAS-II *is paraconsistent.*

To verify this proposition, it is only necessary to check the notion of justification in DAS-II. Justification in DAS-II is defined on top of "strictly defeat". In the development of DAS-I and DAS-II, we intentionally keep this unchanged and at the same time, enrich the underlying notions of "conflicts" and "defeat". This results in an unchanged property at the argument level. That is there are still no two conflicting arguments justified at the same time. Unresolved conflicting arguments are localized in the set of "defensible arguments". Thus, contradiction cannot trivialize the whole system. Consider the argumentation system described in Table 5.6. The arguments found are

$$Arg_1 = \{ r_1 \}$$
$$Arg_2 = \{ r_1, r_2 \}$$
$$Arg_3 = \{ r_1, r_3 \}$$
$$Arg_4 = \{ r_1, r_4 \}$$

Table 5.6: Example showing paraconsistency of DAS-II

EXPERT	KNOWLEDGE
A	$r_1 : \rightarrow stock\_raise$
	$r_2 : stock\_raise \rightarrow customer\_index\_raise$
	$r_3 : stock\_raise \rightarrow \neg customer\_index\_raise$
	$r_4 : stock\_raise \rightarrow stock\_buyer\_nervous$
	$Preference\ Hierarchy = \emptyset$

By definition, Arg_2 and Arg_3 balancely rebuts each other. As the preference hierarchy is empty, Arg_2 and Arg_3 strictly defeat each other. Thus, Arg_2 and Arg_3 cannot be in the set of justified arguments. Arg_1 and Arg_4 are justified as they have no defeaters. This example shows that contradiction (e.g. between Arg_2 and Arg_3) in DAS-II does not upset the whole system.

5.7 An illustrative example

Scenario: In a business re-engineering process, a country-wide manufacturing firm is going to build an autonomous strategical decision support system. The system is designed to provide strategical advice on future directions of the firm based on various economical and social factors. After extensive knowledge engineering efforts, opinions from two domain experts are modelled, as presented in Table 5.1 and Table 5.3. In addition, it was known that the two experts have professional ranking in that A is more professional than B. It was hoped that different perspectives of these two experts

could help to reveal argumentative and critical issues as well as to prevent imprudent decisions.

Qualitative Analysis: Consider the problem of "whether a *new_production_line* should be built ?". Consider a set of rule Arg_1 = { A_2, A_4, A_7, A_3, A_8, A_9, A_{10}, A_1, A_6 }. Arg_1 has the following two splits:

- Split 1 of $Arg_1 = \{A_2, A_4, A_7, A_3, \mathbf{A_9(1)}, A_{10}, A_1, A_6\}$ and
- Split 2 of $Arg_1 = \{A_2, A_4, A_7, A_8, \mathbf{A_9(2)}, A_{10}, A_1, A_6\}$

Note that $\mathbf{A_9(1)}$ and $\mathbf{A_9(2)}$ are just symbols used in here to denote the different versions of A_9 enumerated in the process of spliting. $A_9(1)$ and $A_9(2)$ are variants of A_9 enumerated as follow:

- $A_9\mathbf{(1)}$ *denotes* A_9 : *crime_rate_lower* $\rightarrow$ *emmigration_rate_lower*
- $A_9\mathbf{(2)}$ *denotes* A_9 : *crime_rate_lower* $\rightarrow$ *immigration_rate_lower*

By definition, Arg_1 is an argument. There are three arguments in conflicts with Arg_1. They were:

- $Arg_2 = \{A_5, A_7\}$
- $Arg_3 = \{B_2, B_1, B_4, B_9\}$
- $Arg_4 = \{B_5, B_6\}$

Arg_2 is in the same agent of Arg_1 and is defeated by $Arg_5 = \{A_3, A_8, A_9, A_{10}\}$ as $< A_3, A_5 > \in \tau$ of KB_A. Arg_3 and Arg_4 are derived from another argument. Arg_3 is rebut by Arg_1 and must be resolved by preference hierarchy over agents. Arg_1 defeats Arg_3 as KB_A is assumed stronger than KB_B. Arg_4 undercuts Arg_1. However, Arg_4 is defeated by $Arg_6 = \{B_8, B_7\}$.

If thinning conflict was not considered, $Arg_5, Arg_3, Arg_1 \in T_0(KB_A \cup KB_B)$ and $Arg_1 \in T_1(KB_A \cup KB_B)$. Thus, *new_production_line* would be suggested. If it was considered, B_1 (i.e. from expert B) thinned Arg_1 (i.e. from expert A), *new_production_line* would not be suggested. As a whole, there are no other derivations which could suppress the thinning factors *competition_grow* and yield *new_production_line* at the same time.

By the above reasoning, "new_production_line" would not be suggested. If we removed thinning conflicts detection, "new_production_line" would then be derivable which clearly undermined the opinions of B. This would be imprudent and would be against our basic assumption of skeptical view.

5.8 Chapter summary

In this chapter, we have proposed a distributed argumentation framework, DAS-II, which is a successor to DAS-I. DAS-II is able to capture, analyse and resolve not only classical conflicts like rebuttal and undercut but also thinning. Moreover, it has the ability of "reasoning about cases" which significantly enrich its usage when knowledge is unfocused.

Chapter 6

Evaluation

6.1 Introduction

In this chapter, we shall evaluate the two argumentation frameworks, DAS-I and DAS-II. The evaluation is intended to show that both of them are practical and useful. Furthermore, we show that DAS-II is superior than DAS-I.

There are two tests for both DAS-I and DAS-II.

1. Katsumi Inoue's problems
2. Sherlock Holems' problems

The first set is from Katsumi Inoue's problem sets[1]. This set of testing data concentrates on various aspects of non-monotonic reasoning systems. It evaluates how a framework reasoning under inconsistent, incomplete and indefinite information with respect to stable semantics. The proposed answer is from Inoue's MGTP stable semantics theorem prover. Also this test illustrates how DAS-I and DAS-II resemble and differ from stable semantics.

The second set is a scenario based problem from a detective story of Sherlock Holmes. This test illustrates the capability of DAS-I and DAS-II in handling interrelated indefinite and inconsistent information in a distributed setting. As there are no other existing frameworks with this capability, comparisons are mainly done between DAS-I and DAS-II.

Moreover, the scenario based test also focuses on reasoning under incomplete, contradictory and/or indefinite information. For reasoning under contradictory information, the evaluation focus on conflict toleration through paraconsistency and conflict resolution through priority.

[1]See the following web page http://ai.tutics.tut.ac.jp/~inoue

Table 6.1: Entailment across different frameworks

FRAMEWORKS	ENTAILMENTS
DAS-I	A proposition p is entailed by a system if it is in its justified set.
DAS-II	A proposition p is entailed by a system if it is in its justified set.
Stable Semantics	A proposition p is entailed by a system if it is in every answer sets of the system.

This chapter is organized as follows: In Section 6.2, we describe our testing methodology. Base on that, DAS-I and DAS-II are tested and the results are presented in Section 6.3 and Section 6.4, respectively. Analysis of the results are given in Section 6.5. Finally, a summary is drawn in Section 6.6.

6.2 Methodology

In the evalutaion process, the following reasoning systems are involved: DAS-I, DAS-II and Stable Semantics. In the evaluation:

1. A common criteria are used to evaluate DAS-I and DAS-II against other frameworks.
2. We define a higher level "entailment" see Table 6.1.
3. Differences between these frameworks are compared through "entailment".

For each test case, we determine the arguments, conflicts and conclusion in it according to DAS-I and DAS-II. Conclusions are compared to the expected answer of the reference framework[2]. The resemblances and differences of the systems under compared are highlighted at the end of each test.

DAS-I and DAS-II are propositional frameworks without variables. Several well known examples, e.g. Katsumi Inout's problem set, are formulated in propositional theory with variables. In order to make problems suitable for DAS-I and DAS-II, the original set is based on its Herbrand universe and instantiated. Redundant rules are then removed.

A few points on Katsumi Inoue's problem set must be mentioned. Katsumi Inoue proposed the set of 34 examples problems for knowledge based systems. The problem set is collected from several works by Ken Satch, Kowalski, Gelfond, Lifschitz, Inoue,

[2]The standard answers provided by the test cases.

Satch, Arima and Iwayama. The first 28 problems extensively test different capabilities of knowledge based systems including capabilities of handling various forms of inconsistencies, indefiniteness and incompleteness. The last 6 problems are scenarios-based problems which introduced no new technicalities beyond the first 28. In this chapter, we concentrate on the former 28 problems. Notice that among them, there are three duplicated problems and five inapplicable problems[3]. Thus, we only consider 20 problems (see Section 6.3).

The second problem set is derived from the story "Silver Blaze" of Sir Arthur Conan Doyle's famous novel sequels "Sherlock Holmes". The story plot of "Silver Blaze" is simple for analysis. Yet, arguments employed by Sherlock Holmes shown archetypal commonsense reasoning. Moreover, the articulation between Holmes and Inspector Gregory reflect the essence of distributed argumentation. See Appendix C.2 for details.

6.3 DAS I

6.3.1 Inoue's Benchmark problems

1. **The Archetypal Even Loop**

   The interesting part of this problem is p and $\sim p$ are separated by even number of steps and thus called an even loop.

   $r_1 : \sim q \rightarrow p$
   $r_2 : \sim p \rightarrow q$

   Arguments : $Arg_1 = \{ r_1 \}$, $Arg_2 = \{ r_2 \}$.
   Conflicts : Arg_1 undercuts Arg_2. Arg_2 undercuts Arg_1. Arg_1 defeats Arg_2. Arg_2 defeats Arg_1.
   Conclusion : Arg_1 and Arg_2 are both defensible but not justified. Thus, nothing is entailed.

2. **Reasoning by Cases**

   $r_1 : \sim q \rightarrow p$
   $r_2 : \sim p \rightarrow q$
   $r_3 : p \rightarrow r$
   $r_4 : q \rightarrow r$

[3]Those problems are inapplicable as our framework does not allow clauses without conclusions, i.e. non-conclusive.

Arguments : $Arg_1 = \{ r_1 \}$, $Arg_2 = \{ r_2 \}$, $Arg_3 = \{ r_1, r_3 \}$, $Arg_4 = \{ r_2, r_4 \}$
Conflicts : Arg_1 undercuts and defeats Arg_2 and Arg_4, Arg_2 undercuts and defeats Arg_1 and Arg_3.
Conclusion : All arguments are in the defensible set.

3. **Stratification**

$$r_1 : \sim q \rightarrow p$$
$$r_2 : \sim r \rightarrow q$$

Arguments : $Arg_1 = \{ r_1 \}$, $Arg_2 = \{ r_2 \}$
Conflicts : Arg_2 undercuts and defeats Arg_1.
Conclusion : Arg_2 is justified. q is entailed.

4. **Tautological Loop**

$$r_1 : \sim q \rightarrow p$$
$$r_2 : q \rightarrow q$$

Arguments : $Arg_1 = \{ r_1 \}$
Conflicts : None.
Conclusion : Arg_1 is justified. p is entailed.

5. **The Archetypal Odd Loop**

$$r_1 : \sim p \rightarrow p$$

Arguments : $Arg_1 = \{ r_1 \}$
Conflicts : Arg_1 is defeated $\emptyset$.
Conclusion : None.

6. **Conditional Odd Loop**

$$r_1 : \sim p \wedge q \rightarrow p$$

Arguments : None.
Conflicts : None.
Conclusion : None.

7. **Three-Loop**

$$r_1 : \sim q \rightarrow p$$
$$r_2 : \sim r \rightarrow q$$
$$r_3 : \sim p \rightarrow r$$

Arguments : $Arg_1 = \{ r_1 \}$, $Arg_2 = \{ r_2 \}$, $Arg_3 = \{ r_3 \}$
Conflicts : Arg_1 defeats and undercuts Arg_3. Arg_3 defeats and undercuts Arg_2. Arg_2 defeats and undercuts Arg_1.
Conclusion : Arg_1, Arg_2 and Arg_3 are all defensible. p, q, r are defensible.

8. **Odd and Even Loop**

$r_1 : \sim r \rightarrow r$
$r_2 : q \rightarrow r$
$r_3 : \sim q \rightarrow p$
$r_4 : \sim p \rightarrow q$

Arguments : $Arg_1 = \{ r_1 \}$, $Arg_2 = \{ r_3 \}$, $Arg_3 = \{ r_4 \}$, $Arg_4 = \{ r_4, r_2 \}$
Conflicts : Arg_1 undercuts and defeats Arg_1. Arg_2 undercuts and defeats Arg_3 and Arg_4. Arg_3 undercuts and defeats Arg_2. Arg_4 undercuts and defeats Arg_1 and Arg_2
Conclusion : p, q, r are all defensible.

9. **Conditional Odd and Even Loop**

$r_1 : p \wedge \sim r \rightarrow r$
$r_2 : \sim q \rightarrow p$
$r_3 : \sim p \rightarrow q$

Arguments : $Arg_1 = \{ r_2 \}$, $Arg_2 = \{ r_3 \}$, $Arg_3 = \{ r_2, r_1 \}$
Conflicts : Arg_1 undercuts and defeats Arg_2. Arg_2 undercuts and defeats Arg_1. Arg_3 undercuts and defeats Arg_2.
Conclusion : p, q, r are all defensible.

10. **Two and Three Loop**

$r_1 : \sim q \rightarrow p$
$r_2 : \sim r \rightarrow q$
$r_3 : \sim p \wedge \sim q \rightarrow r$

Arguments : $Arg_1 = \{ r_1 \}$, $Arg_2 = \{ r_2 \}$, $Arg_3 = \{ r_3 \}$
Conflicts : Arg_2 undercuts and defeats Arg_1. Arg_3 undercuts and defeats Arg_2. Arg_1 undercuts and defeats Arg_2.
Conclusion : p, q, r are all defensible.

11. **The Law of Exclusive Middle**

$r_1 : p \rightarrow q$
$r_2 : \rightarrow p \vee \neg p$

Arguments : None.
Conflicts : None.
Conclusion : None.

12. **Incoherent Even Loop**

$r_1 : \sim p \rightarrow q$
$r_2 : \sim q \rightarrow p$
$r_3 : p \rightarrow q$
$r_4 : q \rightarrow p$

Arguments : $Arg_1 = \{ r_1 \}$, $Arg_2 = \{ r_2 \}$, $Arg_3 = \{ r_1, r_4 \}$, $Arg_4 = \{ r_2, r_3 \}$
Conflicts : Arg_1 undercuts and defeats Arg_2 and Arg_4. Arg_2 undercuts and defeats Arg_1 and Arg_3.
Conclusion : All p, q are in defensible input mode.

13. **Inclusive Disjunction**

$r_1 : \rightarrow p \vee q$
$r_2 : p \rightarrow q$
$r_3 : q \rightarrow p$

Arguments : None.
Conflitcs : None.
Conclusion : None.

14. **Minimality**

$r_1 : \rightarrow p \vee q$
$r_2 : q \rightarrow p$

Arguments : None.
Conflicts : None.
Conclusion : None.

15. **Normal Default**

r_1 : *bird(polly)* $\wedge \sim \neg flies(polly) \rightarrow$ *flies(polly)*
r_2 : *bird(tweety)* $\wedge \sim \neg flies(tweety) \rightarrow$ *flies(tweety)*
r_3 : *penguin(polly)* $\rightarrow \neg flies(polly)$
r_4 : *penguin(tweety)* $\rightarrow \neg flies(tweety)$
r_5 : *penguin(polly)* $\rightarrow$ *bird(polly)*
r_6 : *penguin(tweety)* $\rightarrow$ *bird(tweety)*
r_7 : $\rightarrow$ *bird(polly)*
r_8 : $\rightarrow$ *penguin(tweety)*

Arguments : $Arg_1 = \{ r_7 \}$, $Arg_2 = \{ r_8 \}$, $Arg_3 = \{ r_7, r_1 \}$, $Arg_4 = \{ r_8, r_4 \}$, $Arg_5 = \{ r_8, r_6 \}$, $Arg_6 = \{ r_8, r_6, r_2 \}$
Conflicts : Arg_4 rebuts and undercuts Arg_6. Arg_6 rebuts Arg_4. Arg_4 defeats Arg_6.
Conclusion : $Arg_1, Arg_2, Arg_3, Arg_4, Arg_5$ are justified. Arg_6 is defeated. *bird(polly), penguin(tweety), flies(polly),* $\neg flies(tweety)$, *bird(tweety)* are entailed.

16. **Nixon Diamond**

r_1 : *quaker(nixon)* $\wedge \sim$ *ab_quaker(nixon)* $\rightarrow$ *dove(nixon)*
r_2 : *republican(nixon)* $\wedge \sim$ *ab_republican(nixon)* $\rightarrow$ *hawk(nixon)*
r_3 : $\rightarrow$ *quaker(nixon)*
r_4 : $\rightarrow$ *republican(nixon)*
r_5 : *hawk(nixon)* $\rightarrow$ *ab_quaker(nixon)*
r_6 : *dove(nixon)* $\rightarrow$ *ab_republician(nixon)*

Arguments : $Arg_1 = \{ r_3 \}$, $Arg_2 = \{ r_4 \}$, $Arg_3 = \{ r_3, r_1 \}$, $Arg_4 = \{ r_3, r_1, r_6 \}$, $Arg_5 = \{ r_4, r_2 \}$, $Arg_6 = \{ r_4, r_2, r_5 \}$
Conflicts : Arg_4 undercuts and defeats Arg_6 and Arg_5. Arg_6 undercuts and defeats Arg_4 and Arg_3.
Conclusion : Arg_1, Arg_2 are justified. $Arg_3, Arg_4, Arg_5, Arg_6$ are defensible. *quaker(nixon), republican(nixon)* are entailed.

17. **Barber's Non Paradox**

r_1 : *barber(noel)* $\wedge$ *citizen(casanova)* $\wedge \sim$ *shaves(casanova, casanova)* $\rightarrow$ *shaves(noel, casanova)*
r_2 : *barber(casanova)* $\wedge$ *citizen(noel)* $\wedge \sim$ *shaves(noel, noel)* $\rightarrow$ *shaves(casanova, noel)*
r_3 : *barber(noel)* $\wedge$ *citizen(noel)* $\wedge \sim$ *shaves(noel, noel)* $\rightarrow$ *shaves(noel,noel)*
r_4 : *barber(casanova)* $\wedge$ *citizen(casanova)* $\wedge \sim$ *shaves(casanova, casanova)*

$\rightarrow$ *shaves(casanova,casanova)*
r_5 : *barber(noel)* $\rightarrow$ *shaves(noel,noel)*
r_6 : *barber(casanova)* $\rightarrow$ *shaves(casanova,casanova)*
r_7 : $\rightarrow$ *barber(noel)*
r_8 : $\rightarrow$ *citizen(casanova)*
r_9 : *barber(noel)* $\rightarrow$ *citizen(noel)*

Arguments : $Arg_1 = \{ r_7 \}$, $Arg_2 = \{ r_8 \}$, $Arg_3 = \{ r_7, r_5 \}$[4], $Arg_4 = \{ r_7, r_8, r_1 \}$, $Arg_5 = \{ r_7, r_9 \}$, $Arg_6 = \{ r_7, r_9, r_3 \}$
Conflicts : Arg_3 undercuts and defeats Arg_6.
Conclusion : $Arg_1, Arg_2, Arg_3, Arg_4, Arg_5$ are justified. Arg_6 is defeated. *barber(novel), citizen(casanova), shaves(noel, noel), shaves(noel,casanova), citizen(noel)* are entailed.

18. **Disjunctive and Closed World Assumption**

r_1 : $\rightarrow p(a) \vee p(b)$
r_2 : $dom(a) \wedge \sim p(a) \rightarrow \neg p(a)$
r_3 : $dom(b) \wedge \sim p(b) \rightarrow \neg p(b)$
r_4 : $\rightarrow dom(a)$
r_5 : $\rightarrow dom(b)$

Arguments : $Arg_1 = \{ r_4 \}$, $Arg_2 = \{ r_5 \}$, $Arg_3 = \{ r_4, r_2 \}$, $Arg_4 = \{ r_5, r_3 \}$, $Arg_5 = \{ r_4, r_2, r_1 \}$, $Arg_6 = \{ r_5, r_3, r_1 \}$
Conflicts : Arg_5 undercuts, rebuts and defeats Arg_6. Arg_6 undercuts, rebuts and defeats Arg_5. Arg_5 undercuts and defeats Arg_4. Arg_6 undercuts and defeats Arg_3.
Conclusion : Arg_1, Arg_2 are justified. Arg_3, Arg_4, Arg_5 and Arg_6 are defensible. *dom(a), dom(b)* are entailed.

19. **Naming Closed World Assumption**

r_1 : $p \rightarrow q$
r_2 : $\neg p \rightarrow q$
r_3 : $\rightarrow \neg q$
r_4 : $assume(\neg p) \wedge \sim p \rightarrow \neg p$
r_5 : $\sim \neg assume(\neg p) \rightarrow assume(\neg p)$
r_6 : $\sim assume(\neg p) \rightarrow \neg assume(\neg p)$

[4]Rules here are in a deliberate order to reflect derivation steps.

Arguments : $Arg_1 = \{ r_3 \}$, $Arg_2 = \{ r_5 \}$, $Arg_3 = \{ r_6 \}$, $Arg_4 = \{ r_5, r_4 \}$, $Arg_5 = \{ r_5, r_4, r_2 \}$
Conflicts : Arg_1 and Arg_5 rebut and defeat each other. Arg_2 and Arg_3 undercut and defeat each other. Arg_3 undercuts and defeats Arg_4.
Conclusion : None.

20. **Nonmonotonic Assumption-based Truth Maintenance System**

$r_1 : b \rightarrow p$
$r_2 : \sim p \wedge a \rightarrow q$
$r_3 : \sim q \rightarrow p$
$r_4 : \sim out\_a \rightarrow a$
$r_5 : \sim a \rightarrow out\_a$
$r_6 : \sim out\_b \rightarrow b$
$r_7 : \sim b \rightarrow out\_b$

Arguments : $Arg_1 = \{ r_3 \}$, $Arg_2 = \{ r_4 \}$, $Arg_3 = \{ r_5 \}$, $Arg_4 = \{ r_6 \}$, $Arg_5 = \{ r_7 \}$, $Arg_6 = \{ r_4, r_2 \}$, $Arg_7 = \{ r_6, r_1 \}$
Conflicts : Arg_2 and Arg_3 undercut and defeat each other. Arg_4 and Arg_5 undercut and defeat each other. Arg_1 and Arg_7 undercut and defeat Arg_6. Arg_5 undercuts and defeats Arg_7.
Conclusion : None.

Table 6.2 summaries how DAS-I compares to stable semantics. Among the 20 tests, DAS-I scores 13. The failed seven tests can be classified into 3 groups:

1. **Inclusive Disjunctive, Minimality** :
   This group requires "reasoning about cases" capability which is provided in DAS-II.

2. **Reasoning by Cases** :
   The answer in this test is arguable according to our rationale in designing DAS-I. In this test, it is possible to separate conflicting knowledge into two consistent subsets. Inoue called the process of integrating individual reasoning results from the two subsets as "reasoning by cases". A "case" is a consistent subset rather than the result of disjunctions. Thus, "reasoning by cases" is actually reasoning by consistent subsets.

   It can be claimed that "$\sim p \rightarrow q$" As such, these two "cases" are indeed conflicting in DAS-I. It is thus improper to draw further results from conflicting basis in DAS-I.

Table 6.2: Summary of DAS-I on Katsumi Inoue's problem set

TESTS	AGREED WITH EXPECTED RESULT?
The Archetypel Even Loop	Yes.
Stratification	Yes.
Tautological Loop	Yes.
The Archetypal Odd Loop	Yes.
Conditional Odd Loop	Yes.
Three-Loop	Yes.
The Law of Exclusive Middle	Yes.
Incoherent Even Loop	Yes.
Normal Default	Yes.
Nixon Diamond	Yes.
Barber's Non Paradox	Yes.
Disjunctive and Closed World Assumption	Yes.
Nonmonotonic Assumption-based Truth Maintenance System	Yes.
Inclusive Disjunctive	No.
Minimality	No.
The Reasoning by Cases	No.
Odd and Even Loop	No.
Conditional Odd and Even Loop	No.
Two and Three Loop	No.
Naming Closed World Assumption	No.

3. **Odd and Even Loop, Conditional Odd and Even Loop, Two and Three Loop, Naming Closed World Assumption** :
Stable semantics resolves conflicts by preferring one of the conflicting part. DAS-I resolves conflicts by leaving both conflicting part alone if conflicts involved are undercut or are without priority for resolution. This difference accounts for why DAS-I does not entail stable semantics' results.

6.3.2 Sherlock Holmes' problems

1. **Straker's Double Life**

In this scenario, Holmes reasoned on Whether Straker had a double life [5]. Firstly, he had the following facts:

$r_1 : \rightarrow carry(straker,bill)$
$r_2 : \rightarrow man(straker)$
$r_3 : \rightarrow name(bill,other)$
$r_4 : \rightarrow usage(bill,dresses)$

Moreover, he had the following commonsense rules:

$r_5 : carry(X,bill) \rightarrow belong(bill,X) \vee belong(bill,other)$
$r_6 : \sim \neg normal(X) \wedge man(X) \rightarrow normal(X)$
$r_7 : normal(X) \wedge carry(X,bill) \rightarrow \neg belong(X,other)$
$r_8 : belong(O,X) \wedge \sim name(O,X) \rightarrow double\_identity(X)$
$r_9 : usage(bill,dresses) \rightarrow for(bill,lady)$
$r_{10} : carry(O,X) \wedge for(O,Y) \rightarrow affliate(X,Y)$
$r_{11} : double\_identity(X) \wedge man(X) \wedge affliate(X,lady) \rightarrow double\_life(X,lady)$
$r_{12} : double\_identity(X) \wedge woman(X) \wedge affliate(X,gentleman) \rightarrow double\_life(X,gentleman)$
$r_{13} : \rightarrow servant(straker)$
$r_{14} : \rightarrow huge\_expense(straker)$
$r_{15} : huge\_expense(X) \rightarrow to\_earn(X)$
$r_{16} : huge\_expense(X) \wedge little\_money(X) \rightarrow crime\_motive(X)$
$r_{17} : to\_earn(X) \rightarrow to\_work(X) \vee to\_crime(X)$
$r_{18} : servant(X) \wedge to\_work(X) \rightarrow little\_money(X)$
$r_{19} : to\_crime(X) \rightarrow crime\_motive(X)$
$r_{20} : double\_life(X,lady) \rightarrow indecent(X)$

[5] See Appendix C.1. for original text excerpt.

r_{21} : *crime_motive(X)* $\wedge$ *indecent(X)* $\rightarrow$ *suspect(X)*

Let the union of the above knowledge be K. Instantiating K on K's Herbrand Universe, we have the grounded version of K, denoted as K_g, as follows:

r_1 : $\rightarrow$ *carry(straker,bill)*
r_2 : $\rightarrow$ *man(straker)*
r_3 : $\rightarrow$ *name(bill,other)*
r_4 : $\rightarrow$ *usage(bill,dresses)*
r_5 : *carry(straker,bill)* $\rightarrow$ *belong(bill,straker)* $\vee$ *belong(bill,other)*
r_6 : $\sim \neg$*normal(straker)* $\wedge$ *man(straker)* $\rightarrow$ *normal(straker)*
r_7 : *normal(straker)* $\wedge$ *carry(straker,bill)* $\rightarrow \neg$*belong(bill, other)*
r_8 : *belong(bill,straker)* $\wedge \sim$ *name(bill, straker)* $\rightarrow$ *double_identity(straker)*
r_9 : *belong(bill,other)* $\wedge \sim$ *name(bill, other)* $\rightarrow$ *double_identity(other)*
r_{10} : *usage(bill,dresses)* $\rightarrow$ *for(bill,lady)*
r_{11} : *carry(straker,bill)* $\wedge$ *for(bill,lady)* $\rightarrow$ *affliate(straker,lady)*
r_{12} : *double_identity(straker)* $\wedge$ *man(straker)* $\wedge$ *affliate(straker,lady)* $\rightarrow$ *double_life(straker,lady)*
r_{13} : *double_identity(straker)* $\wedge$ *woman(straker)* $\wedge$ *affliate(straker,gentleman)* $\rightarrow$ *double_life(straker,gentleman)*
r_{14} : $\rightarrow$ *servant(straker)*
r_{15} : $\rightarrow$ *huge_expense(straker)*
r_{16} : *huge_expense(straker)* $\rightarrow$ *to_earn(straker)*
r_{17} : *huge_expense(straker)* $\wedge$ *little_money(straker)* $\rightarrow$ *crime_motive(straker)*
r_{18} : *to_earn(straker)* $\rightarrow$ *to_work(straker)* $\vee$ *to_crime(straker)*
r_{19} : *servant(straker)* $\wedge$ *to_work(straker)* $\rightarrow$ *little_money(straker)*
r_{20} : *to_crime(straker)* $\rightarrow$ *crime_motive(straker)*
r_{21} : *double_life(straker,lady)* $\rightarrow$ *indecent(straker)*
r_{22} : *crime_motive(straker)* $\wedge$ *indecent(straker)* $\rightarrow$ *suspect(straker)*

Let the argumentation system $AS =< Ags, Pg >$ where $Ags =< Ag >$, $Pg =< \emptyset >$ and $Ag =< r_1, \cdots, r_{13} >$. The following major arguments are determined:

$Arg_1 = \{ r_2, r_6, r_1, r_7, r_5, r_8 \}$
$Arg_2 = \{ r_4, r_{10}, r_1, r_{11} \}$
$Arg_3 = Arg_1 + Arg_2 + \{ r_{12} \}$

There are no conflicts and all arguments are justified. The conclusions of this knowledge set are *carry(straker, bill)* , *man(straker)* , *name(bill, anon)* , *usage(bill, dresses)* , *normal(straker)* , $\neg belong(bill, other)$, *belong(bill, straker)*, *double_identity(straker)*, *for(bill, lady)*, *affliate(straker, lady)*, *double_life(straker, lady)*, *servant(straker)* .

Attentions should be drawn on how r_7 and r_5 work together through *certain*() in DAS-I. r_7 is a piece of indefinite information. There are no existing argumentation frameworks, to the author's best knowledge, capable of reasoning in this way.

2. **Poison Stable Boy**

In this scenario, two experts, namely Inspector Gregory and Sherlock Holmes, are articulating on the problem "poison stable boy"[6]. The problem formulated in plain propositional logic with variables is as follows:

EXPERTS	KNOWLEDGE
Gregory	$a_1 : \rightarrow suspect(simpson)$
	$a_2 : \rightarrow passby(gypies)$
	$a_3 : suspect(X) \rightarrow injure(X,horse)$
	$a_4 : injure(X,horse) \rightarrow take\_out(X,horse)$
	$a_5 : take\_out(X,horse) \rightarrow posion(stable\_boy)$
	$a_6 : poision(X) \rightarrow \neg awaken(X)$
	$a_7 : \sim \neg awaken(stable\_boy) \rightarrow \neg take\_out(Y, horse)$
	$a_8 : suspect(X) \rightarrow want(X,benefits)$
	$a_9 : want(X,benefits) \wedge passby(gypies) \rightarrow meet(X,gypies)$
	$a_{10} : meet(X,gypies) \rightarrow \neg kill\_there(X, horse)$
Holmes	$b_1 : \rightarrow suspect(simpson)$
	$b_2 : \rightarrow passby(gypies)$
	$b_3 : suspect(X) \rightarrow injure(X,horse)$
	$b_4 : injure(X,horse) \rightarrow take\_out(X,horse) \vee kill\_there(X,horse)$
	$b_5 : take\_out(X,horse) \rightarrow posion(stable\_boy)$
	$b_6 : poision(X) \rightarrow \neg awaken(X)$
	$b_7 : \sim \neg awaken(stable\_boy) \rightarrow \neg take\_out(Y, horse)$
	$Preference\ Hierarchy = \emptyset$

To facilitate reasoning in the propositional framework DAS-I, we instantiated the aforesaid knowledge, according to the Herbrand universe of the problem. This results in the following rules:

[6]See Appendix C.2.

EXPERTS	KNOWLEDGE
Gregory	a_1 : $\rightarrow$ *suspect(simpson)*
	a_2 : $\rightarrow$ *passby(gypies)*
	a_3 : *suspect(simpson)* $\rightarrow$ *injure(simpson,horse)*
	a_4 : *injure(simpson,horse)* $\rightarrow$ *take_out(simpson,horse)*
	a_5 : *take_out(simpson,horse)* $\rightarrow$ *posion(stable_boy)*
	a_6 : *poision(stable_boy)* $\rightarrow$ $\neg$*awaken(stable_boy)*
	a_7 : $\sim \neg$*awaken(stable_boy)* $\rightarrow$ $\neg$*take_out(simpson, horse)*
	a_8 : *suspect(simpson)* $\rightarrow$ *want(simpson,benefits)*
	a_9 : *want(simpson,benefits)* $\wedge$ *passby(gypies)* $\rightarrow$ *meet(simpson,gypies)*
	a_{10} : *meet(simpson,gypies)* $\rightarrow$ $\neg$*kill_there(simpson, horse)*
Holmes	b_1 : $\rightarrow$ *suspect(simpson)*
	b_2 : $\rightarrow$ *passby(gypies)*
	b_3 : *suspect(simpson)* $\rightarrow$ *injure(simpson,horse)*
	b_4 : *injure(simpson,horse)* $\rightarrow$ *take_out(simpson,horse)* $\vee$ *kill_there(simpson,horse)*
	b_5 : *take_out(simpson,horse)* $\rightarrow$ *posion(stable_boy)*
	b_6 : *poision(stable_boy)* $\rightarrow$ $\neg$*awaken(stable_boy)*
	b_7 : $\sim \neg$*awaken(stable_boy)* $\rightarrow$ $\neg$*take_out(simpson, horse)*
	Preference Hierarchy $= \emptyset$

Let argument system $AS =< Ags, Pg >$ where $Ags =< Ag_G, Ag_H >$, $Pg =< \emptyset >$, $Ag_G = \{a_1, \cdots, a_{10}\}$ and $Ag_H = \{b_1, \cdots, b_7\}$. Ag_G and Ag_H denote argumentation agent of Gregory and Holmes, respectively. The following major arguments are found:

$$Arg_1 = \{\ a_1,\ a_2,\ a_3,\ a_4,\ a_5,\ a_6\ \}$$
$$Arg_2 = \{\ a_7\ \}$$
$$Arg_3 = \{\ a_1,\ a_8,\ a_9,\ a_{10}\ \}$$
$$Arg_4 = \{\ b_1,\ b_2,\ b_3,\ b_4,\ b_7\ \}$$

Arg_1 from Ag_G is thinned by the rule b_4 from Ag_H. Arg_3 from Ag_G defeats b_4 for Arg_1. To see how b_4 actually affects the results, it is necessary to see that Arg_4 is an argument similar to Arg_1 but with completely different conclusions. The replacement of b_4 with a_4 results in another interpretation.

6.4 DAS II

6.4.1 Inoue's benchmark problems

In this section, we will not repeat the problem in the following tests unless the results in here is different from Section 6.3.1. Their original formulation can be found in the corresponding parts of Section 6.3.1.

1. **The Archetypal Even Loop**

   The same as Section 6.3.1.

2. **Reasoning by Cases**

   The same as Section 6.3.1.

3. **Stratification**

   The same as Section 6.3.1.

4. **Tautological Loop**

   The same as Section 6.3.1.

5. **The Archetypal Odd Loop**

   $$r_1 : \sim p \rightarrow p$$

   Arguments : None.
   Conflicts : None.
   Conclusion : None.

6. **Conditional Odd Loop**

   The same as Section 6.3.1.

7. **Three-Loop**

   The same as Section 6.3.1.

8. **Odd and Even Loop**

   The same as Section 6.3.1.

9. **Conditional Odd and Even Loop**

   The same as Section 6.3.1.

10. **Two and Three Loop**

    The same as Section 6.3.1.

11. **The Law of Exclusive Middle**

$r_1 : p \rightarrow q$
$r_2 : \rightarrow p \vee \neg p$

The same as Section 6.3.1. Note that $\{ r_1 , r_2 \}$ is not an argument as there does not exist a proposition p which is entailed by all the splits derived from itself.

12. **Incoherent Even Loop**

The same as Section 6.3.1.

13. **Inclusive Disjunction**

$r_1 : \rightarrow p \vee q$
$r_2 : p \rightarrow q$
$r_3 : q \rightarrow p$

Arguments : $Arg_1 = \{ r_1 , r_2 \}$, $Arg_2 = \{ r_1, r_3 \}$
Conflicts : None.
Conclusions : Arg_1 and Arg_2 are justified. p and q are entailed.

14. **Minimality**

$r_1 : \rightarrow p \vee q$
$r_2 : q \rightarrow p$

Arguments : $Arg_1 = \{ r_1, r_2 \}$
Conflicts : None.
Conclusions : Arg_1 is justified. p is entailed.

15. **Normal Default**

The same as Section 6.3.1.

16. **Nixon Diamond**

The same as Section 6.3.1.

17. **Barber's Paradox**

The same as Section 6.3.1.

18. **Disjunctive and Closed World Assumption**

The same as Section 6.3.1.

Table 6.3: Summary of DAS-II on Katsumi Inoue's problem set

TESTS	AGREED WITH EXPECTED RESULT?
The Archetypel Even Loop	Yes.
Stratification	Yes.
Tautological Loop	Yes.
The Archetypal Odd Loop	Yes.
Conditional Odd Loop	Yes.
Three-Loop	Yes.
The Law of Exclusive Middle	Yes.
Incoherent Even Loop	Yes.
Normal Default	Yes.
Nixon Diamond	Yes.
Barber's Non Paradox	Yes.
Disjunctive and Closed World Assumption	Yes.
Nonmonotonic Assumption-based Truth Maintenance System	Yes.
* Inclusive Disjunctive	Yes.
* Minimality	Yes.
The Reasoning by Cases	No.
Odd and Even Loop	No.
Conditional Odd and Even Loop	No.
Two and Three Loop	No.
Naming Closed World Assumption	No.

19. **Naming Closed World Assumption**

    The same as Section 6.3.1.

20. **Nonmonotonic Assumption-based Truth Maintenance System**

    The same as Section 6.3.1.

Table 6.3 summaries how DAS-II compares to stable semantics. DAS-II extends DAS-I for "Inclusive Disjunctive" and "Minimality" tests with respect to stable semantics. Thus, DAS-II is closer to stable semantics than DAS-I for the Inoue's tests.

6.4.2 Sherlock Holmes' problem

The formulation is outlined in Section 6.3.2 and is not repeated here.

1. **Straker's Double Life**

   Let the argumentation system $AS =< Ags, Pg >$ where $Ags =< Ag >$, $Pg =< \emptyset >$ and $Ag =< r_1, \cdots, r_{13} >$. The following major arguments are found:

   $$Arg_1 = \{ r_2, r_6, r_1, r_7, r_5, r_8 \}$$
   $$Arg_2 = \{ r_4, r_{10}, r_1, r_{11} \}$$
   $$Arg_3 = Arg_1 + Arg_2 + \{ r_{12} \}$$
   $$Arg_4 = \{ r_{14}, r_{15}, r_{16}, r_{18}, r_{19}, r_{17}\}$$
   $$Arg_5 = Arg_4 + \{r_{20}, r_{21}, r_{22}\}$$

   Similar to result in Section 6.3.2, there are no conflicts identified. Moreover, All arguments are justified. Differed from Section 6.3.2, the conclusions of this knowledge set is extended to include $suspect(straker)$. $suspect(straker)$, which means "Straker is a suspect.", is actually the critical point of the whole story. This illustrates that DAS-II is significantly more powerful than DAS-I. Without DAS-II's "reasoning about cases" capability, the conclusion cannot be drawn at all.

2. **Poison Stable Boy**

   The same as 6.3.2.

6.5 Analysis

Table 6.2 summaries how DAS-I compares to stable semantics. Among the 20 tests, DAS-I scores 13. The failed seven tests can be classified into 3 groups:

1. **Inclusive Disjunctive, Minimality** :
   This group requires "reasoning about cases" capability which is provided in DAS-II.

2. **Reasoning by Cases** :
   The answer in this test is arguable according to our rationale in designing DAS-I. In this test, it is possible to separate conflicting knowledge into two consistent subsets. Inoue called the process of integrating individual reasoning results from the two subsets as "reasoning by cases". A "case" is actually a consistent subset but not a result of disjunction. Thus, "reasoning by cases" is actually "reasoning by consistent subsets".

It can be argued that "$\sim p \rightarrow q, \sim q \rightarrow p$" is used to represent $\rightarrow p \vee q$. However, this may mix up the meaning of undercut and disjunction. Using disjunction, we can intuitively define three or more possible outcomes, e.g. $\rightarrow p_1 \vee p_2 \vee p_3 \vee \cdots$. It is very difficult if not impossible to do so with undercut. Thus, "$\sim p \rightarrow q, \sim q \rightarrow p$" is not a good alternative for "$\rightarrow p \vee q$".

If $\sim p \rightarrow p$ and $\sim p \rightarrow q$ are interpreted literally, they yield two consistent subsets which undercut each other. Both are not conflict free nor backed up by another consistent subset free from conflicts. Thus, they are not beyond reasonable doubts. By skepticism, we cannot sanction their results.

3. **Odd and Even Loop, Conditional Odd and Even Loop, Two and Three Loop, Naming Closed World Assumption** :
Stable semantics resolves conflicts by preferring one of the conflicting part. DAS-I resolves conflicts by leaving both conflicting part alone if conflicts involved are undercut or are without priority for resolution. This difference accounts for why DAS-I does not entail stable semantics' results.

DAS-II significantly extends DAS-I's result to "Inclusive Disjunctive" test and "Minimality" test. However, there are still five tests deviate from stable semantics. It does not surprise us as the results of them depends on the assumptions of stable semantics.

6.5.1 Possible extension

It is worth noting that four out of the five unsolved problems in DAS-II (see Table 6.3) could be handled "properly in the sense of stable semantics" by the extension proposed by Prakken [Prakken and Sartor, 1997]. Similary, they could be introduced to DAS-II. The extensions are defined as follows:

Definition 38 *Given a set of arguments A, a subset S of A is stable if S does not conflict with itself and for every $a \in A - S$, S defeats a.*

Given an argumentation framework AF and its argument set $Args$, denotes the set of justified, defensible and defeated argument sets as $Args^{+}$, $Args^{0}$ and $Args^{-}$ respectively.

Definition 39 *For an defensible argument set Arg of an argument framework AF, an argument is stably justified if it is justified in every stable subset of Arg.*

We then use this extended notion of "justify" to test the failed five problems as follows:

2 Reasoning by Cases

$r_1 : \sim q \rightarrow p$
$r_2 : \sim p \rightarrow q$
$r_3 : p \rightarrow r$
$r_4 : q \rightarrow r$

Arguments : $Arg_1 = \{ r_1 \}$, $Arg_2 = \{ r_2 \}$, $Arg_3 = \{ r_1, r_3 \}$, $Arg_4 = \{ r_2, r_4 \}$
Conflicts : Arg_1 undercuts and defeats Arg_2 and Arg_4, Arg_2 undercuts and defeats Arg_1 and Arg_3.
Conclusion : All arguments are in the defensible set. Note that $\{Arg_1, Arg_3\}$ and $\{Arg_2, Arg_4\}$ are stable. r is justified as it is justified in both of them.

8 Odd and Even Loop

$r_1 : \sim r \rightarrow r$
$r_2 : q \rightarrow r$
$r_3 : \sim q \rightarrow p$
$r_4 : \sim p \rightarrow q$

Arguments : $Arg_1 = \{ r_1 \}$, $Arg_2 = \{ r_3 \}$, $Arg_3 = \{ r_4 \}$, $Arg_4 = \{ r_4, r_2 \}$
Conflicts : Arg_1 undercuts and defeats Arg_1. Arg_2 undercuts and defeats Arg_3 and Arg_4. Arg_3 undercuts and defeats Arg_2. Arg_4 undercuts and defeats Arg_1 and Arg_2
Conclusion : All arguments are defensible. Only $\{Arg_4\}$ is stable. q, r are then justified.

9 Conditional Odd and Even Loop

$r_1 : p \wedge \sim r \rightarrow r$
$r_2 : \sim q \rightarrow p$
$r_3 : \sim p \rightarrow q$

Arguments : $Arg_1 = \{ r_2 \}$, $Arg_2 = \{ r_3 \}$, $Arg_3 = \{ r_2, r_1 \}$
Conflicts : Arg_1 undercuts and defeats Arg_2. Arg_2 undercuts and defeats Arg_1. Arg_3 undercuts and defeats Arg_2.
Conclusion : All arguments are defensible. Only $\{Arg_2\}$ is stable so q is justified.

10 Two and Three Loop

$r_1 : \sim q \rightarrow p$
$r_2 : \sim r \rightarrow q$
$r_3 : \sim p \wedge \sim q \rightarrow r$

Arguments : $Arg_1 = \{ r_1 \}$, $Arg_2 = \{ r_2 \}$, $Arg_3 = \{ r_3 \}$
Conflicts : Arg_2 undercuts and defeats Arg_1. Arg_3 undercuts and defeats Arg_2. Arg_1 undercuts and defeats Arg_2.
Conclusion : All arguments are defensible. Only $\{Arg_2\}$ is stable so q is justified.

19 **Naming Closed World Assumption**

$r_1 : p \rightarrow q$
$r_2 : \neg p \rightarrow q$
$r_3 : \rightarrow \neg q$
$r_4 : assume(\neg p) \wedge \sim p \rightarrow \neg p$
$r_5 : \sim \neg assume(\neg p) \rightarrow assume(\neg p)$
$r_6 : \sim assume(\neg p) \rightarrow \neg assume(\neg p)$

Arguments : $Arg_1 = \{ r_3 \}$, $Arg_2 = \{ r_5 \}$, $Arg_3 = \{ r_6 \}$, $Arg_4 = \{ r_5, r_4 \}$, $Arg_5 = \{ r_5, r_4, r_2 \}$
Conflicts : Arg_1 and Arg_5 rebut and defeat each other. Arg_2 and Arg_3 undercut and defeat each other. Arg_3 undercuts and defeats Arg_4.
Conclusion : Only the argument set $\{Arg_1, Arg_3\}$ is stable. so $\neg q$ and $\neg assume(\neg p)$ are justified.

Combining the above results with DAS-II's original result we get Table 6.4. This shows that our framework can readily be extended to cover all benchmark problems. Although, the extension is, in our view, against our basic assumption of skepticism, there may still be situations whence such properties are desirable.

6.6 Chapter summary

In this chapter, we evalutated the capabilities of DAS-I and DAS-II. We show that both DAS-I and DAS-II are both good reasoning framework for Katsumi Inoue's tests. Further, we show that DAS-II can deduce more informations/rules than DAS-I. On the Sherlock Holmes' test, DAS-I and DAS-II both fulfill "Poison Stable Boy" test. DAS-II fulfills "Straker's Double Life"test and draws more conclusions than DAS-I. Thus, we shown that the DAS-II is superior.

Table 6.4: Summary of Extended DAS-II on Katsumi Inoue's problem set

TESTS	AGREED WITH EXPECTED RESULT?
The Archetypel Even Loop	Yes.
Stratification	Yes.
Tautological Loop	Yes.
The Archetypal Odd Loop	Yes.
Conditional Odd Loop	Yes.
Three-Loop	Yes.
The Law of Exclusive Middle	Yes.
Incoherent Even Loop	Yes.
Normal Default	Yes.
Nixon Diamond	Yes.
Barber's Non Paradox	Yes.
Disjunctive and Closed World Assumption	Yes.
Nonmonotonic Assumption-based Truth Maintenance System	Yes.
Inclusive Disjunctive	Yes.
Minimality	Yes.
* The Reasoning by Cases	Yes.
* Odd and Even Loop	Yes.
* Conditional Odd and Even Loop	Yes.
* Two and Three Loop	Yes.
* Naming Closed World Assumption	Yes.

Chapter 7

Conclusion

In this thesis, we have studied the problem of reasoning with indefinite, inconsistent and incomplete information in a distributed setting in argumentative style. We reviewed the general aspects of the subject in chapters 2 and 3. We identified logic programming and argumentation as our representation and philosophy frameworks, respectively. In particular, we found that Henry Prakken's strict argumentation could offer the required supports for our purpose.

The main contribution of our work is the proposal of two disjunctive argumentation semantics, namely DAS-I and DAS-II. They are novel in the following aspects:

1. **Disjunctive**
   Our argumentation frameworks are disjunctive and allows indefinite knowledge to be modelled in a prioritized framework. To our best knowledge, this is the first attempt in argumentation theory to combine the both.

2. **Thinning**
   We identified a new kind of conflicts known as thinning in a distributed knowledge environment. Thinning concerns non-complimentary conflicts between distributed knowledge.

3. **Integrated**
   DAS-I and DAS-II were designed with an integrated approach for handling indefinite, inconsistent and incomplete information. This is effective as conflicts are inter-related in practice. Moreover, our way in analysing relations between indefinite and inconsistent information in distributed settings is novel.

DAS-I is an extension of Henry Prakken's strict argumentation [Prakken and Sartor, 1997]. We introduced an indefinite information reasoning capability to the original Prakken's framework. In doing so, we managed not to ruin the

desirable features of Prakken's framework, e.g. the sound and complete proof procedure which are inherited from Phan Minh Dung. By relocating the single knowledge base/agent scenario into a general distributed agents scenario, we showed that there could be non-trivial conflicts between different agents. To resolve this and other conflicts, DAS-I restricts that each indefinite rule can have only one unique interpretation in an argument. The single layered preference hierarchy in Prakken's is extended to two layers to cope with the difference between distributed agents.

The downside of the simplistic approach of DAS-I is that it cannot support "reasoning about cases". Our second framework, DAS-II, was designed to overcome this predicament. As reasoning about cases called for forward reasoning, DAS-II has to adopt an alternative formulation (i.e. different from DAS-I). Base on technique borrowed from logic programming research, we show that DAS-II subsumes DAS-I and incorporates the capability of "reasoning about cases".

It is realized that rebut conflicts are much more complicated when indefinite rule is not restricted. The conflicting parts do not simply rest on two rules. There are no existing study dealing with this kind of subtlety. We proposed a "bearer and initiator" scheme to analyze and solve the problem.

In summary, our goal for an integrated framework for reasoning with indefinite, inconsistent, incomplete information is fulfilled by DAS-I and DAS-II. We have gone into great details in analyzing tangling aspects of indefiniteness and inconsistency in coherence with the treatment of incompleteness.

7.0.1 Possible extension of the present work

1. *Efficient computation method*
   Devising an efficient computation method for DAS-I is strict forward. The same does not hold for DAS-II which employs forward reasoning mechanism. Enumerating approach is plausible only when the problem is near-Horn, i.e. there are not much indefinite information. As both DAS-I and DAS-II are represented in a logic programming language, it is worth investigating whether program transformation techniques are applicable for efficient computation.

2. *Dynamic priority*
   This is trivial for DAS-I as it borrowed many concepts from Prakken's strict argumentation. Adding dynamic priority to DAS-II requires revision of the resolution schemes as the same set of schemes would also be applied on conflicts over priority.

3. *More or less "stable semantics" oriented*
   We concluded that stable semantics gave the most intuitive results among exist-

ing logic programming approaches in Chapter 2. In the evaluation, we find that DAS-II is not fully compatible with stable semantics. It does not surprise us as both DAS-I and DAS-II are based on Prakken's framework. We share the same skeptical view on information as Prakken. The difference is then not because of fault but rationale. Through the investigation, it is clear that skeptical view actually differs from "stable semantics" on finer points on conflicts resolution. Thus, the issues of whether extending DAS-II to be more, or less, "stable semantics" oriented is an interesting topics.

4. *Alternative resolution schemes*
   As we mentioned above, our conflict resolution schemes are essentially biased as all forms of preference did. Studying of alternative resolution schemes, e.g. quantitative measures (like John Fox's approach [Krause P, 1993]) and rhetorical measures (like Chris Reed's approach [Reed *et al.*, 1996]), for conflicts identified in DAS-I and DAS-II would enrich our frameworks.

The End

Bibliography

[Alexy, 1989] Robert Alexy. *A Theory of Legal Argumentation.* Claredon Press, Oxford, 1989.

[Alferes *et al.*, 1994] José Júlio Alferes, Carlos Viegas Damasio, and Luis Moniz Pereira. Top-down query evaluation for well-founded semantics with explicit negation. In A. G. Cohn, editor, *Proceedings of the Eleventh European Conference on Artificial Intelligence*, pages 140–144, Chichester, August8–12 1994. John Wiley and Sons.

[Baral and Gelfond, 1994] Chitta Baral and Michael Gelfond. Logic programming and knowledge representation. *The Journal of Logic Programming*, 19 & 20:73–148, May 1994.

[Barendregt, 1981] Hendrik Pieter Barendregt. *The lambda calculus : its syntax and semantics.* Amsterdam : North Holland, 1981.

[Belnap, 1977] N. D. Belnap. A useful four-valued logic. In J. M. Dunn and G. Epstein, editors, *Modern Uses of Multiple-valued Logic*, pages 8–37. Reidel, Dordrecht, 1977.

[Bowen and Kowalski, 1982] Kenneth A. Bowen and Robert A. Kowalski. Amalgamating language and metalanguage in logic programming. In K. L. Clark and S.-A. Tarnlund, editors, *Logic programming*, volume 16 of *APIC studies in data processing*, pages 153–172. Academic Press, 1982.

[Brewka, 1996] Gerhard Brewka. Well-founded semantics for extended logic programs with dynamic preferences. *Journal of Artificial Intelligence Research*, 4:19–36, 1996.

[Carnap, 1950] Rudolf Carnap. *Logical Foundations of Probability.* University of Chicago Press, Chicago, 1950.

[Clark, 1978] K. L. Clark. Negation as failure. In H. Gallaire and J. Minker, editors, *Logic and Databases*, pages 293–322, New York, 1978. Plenum Press.

[Colmerauer *et al.*, 1973] Alain Colmerauer, Henri Kanoui, Robert Pasero, and Philippe Roussel. Un systéme de communication homme-machine en Français. Rapport, Groupe d'Intelligence Artificielle, Université d'Aix-Marseille II, 1973.

[da Costa, 1963] Newton C. A. da Costa. Calculs propositionnels pour les systemes formels inconsistants. *C. R. de l'Academie des Sciences de Paris*, 257:3790–3793, 1963.

[Davis and Putman, 1960] Martin Davis and Hilary Putman. A computing procedure for quantification theory. *Journal of the ACM*, 7(3):201–215, July 1960.

[Davis, 1993] Martin Davis. First order logic. In J. A. Robinson Dov M. Gabbay, C. J. Hogger, editor, *Handbook of Logic in Arificial Intelligence and Logic Programming*, volume 2, pages 31–65. Clarendon Press, 1993.

[Dung and Son, 1995] P. M. Dung and T. C. Son. Nonmonotonic inheritance, argumentation and logic programming. In V. W. Marek, A. Nerode, and M. Truszczyński, editors, *Proceedings of the 3rd International Conference on Logic Programming and Nonmonotonic Reasoning*, volume 928 of *LNAI*, pages 316–329, Berlin, Jun 1995. Springer.

[Dung and Son, 1996] Phan Minh Dung and Tran Cao Son. An argumentation-theoretic approach to reasoning with specificity. In Luigia Carlucci Aiello, Jon Doyle, and Stuart Shapiro, editors, *Proceedings of the Fifth International Conference on Principles of Knowledge Representation and Reasoning*, pages 506–517, San Francisco, November 5–8 1996. Morgan Kaufmann.

[Dung, 1995] Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77:321–357, 1995.

[Fernández and Lobo, 1993] J. A. Fernández and J. Lobo. A proof procedure for stable theories. Technical Report UMIACS–TR–93–14 and CS–TR–3034, University of Maryland Institute for Advance Computer Studies, College Park, MD 20742, 1993.

[Frege, 1967] Gottlob Frege. Begriffsschrift, eine der arithmetischen nachgebildete formelsprache des reinen denkens. In J. van Heijenoort, editor, *From Frege to Godel: A Source Book in Mathematical Logic 1879-1931*. Harvard University Press, 1967.

[Gelder *et al.*, 1988] A. Van Gelder, K. A. Ross, and J. S. Schlipf. The well-founded semantics for general logic programs. In *Proc. 7th ACM Symp. on Principles of Database Systems*, pages 221–230, Austin, March 1988. to appear in *Journal of the ACM*.

[Gelfond and Lifschitz, 1988] M. Gelfond and V. Lifschitz. The stable semantics for logic programs. In R. Kowalski and K. Bowen, editors, *Proceedings of the 5th international symposium on logic programming*, pages 1070–1080, Cambridge, MA., 1988. MIT Press.

[Gelfond and Lifschitz, 1990] Michael Gelfond and Vladimir Lifschitz. Logic programs with classical negation. In Peter Warren, David H.D.; Szerdei, editor, *Proceedings of the 7th International Conference on Logic Programming (ICLP '90)*, pages 579–597, Jerusalem, June 1990. MIT Press.

[Gelfond and Lifschitz, 1991] Michael Gelfond and Vladimir Lifschitz. Classical Negation in Logic Programs and Disjunctive Databases. *New Generation Computing*, 9, 1991.

[Gordon, 1993] Thomas F. Gordon. The pleadings game; formalizing procedural justice. In *Proceedings of the Fourth International Conference on Artificial Intelligence and Law*, pages 10–19, Amsterdam, July 15-18 1993. ACM Press.

[Hample, 1977] D. Hample. Testing a model of value argument and evidence. *Communication Monographs*, 44(2):106–120, 1977.

[Herbrand, 1967] Jacques Herbrand. Investigations into proof theory. In J. van Heijenoort, editor, *From Frege to Godel: A Source Book in Mathematical Logic 1879-1931*, volume 16, pages 525–581. Harvard University Press, 1967.

[Inoue *et al.*, 1992] Katsumi Inoue, Miyuki Koshimura, and Ryuzo Hasegawa. Embedding negation as failure into a model generation theorem prover. *Lecture Notes in Computer Science*, 607:400–415, 1992.

[Jaskowski, 1969] Stanislaw Jaskowski. Propositional calculus for contradictory deductive systems. *Studia Logica XXIV*, pages 143–157, 1969.

[Kleene, 1952] Stephen Cole Kleene. *Introduction to metamathematics.* New York : Van Nostrand, 1952.

[Kleene, 1967] Stephen Cole Kleene. *Mathematical logic.* John Wiley & Sons, 1967.

[Konolige, 1989] Kurt Konolige. On the Relations between Autoepistemic Logic and Circumscription. In *Proceedings IJCAJ*, pages 1213–1218, 1989.

[Kowalski and Kim, 1991] R. Kowalski and J.-S. Kim. A metalogic programming approach to multi-agent knowledge and belief. In V. Lifschitz, editor, *AI and Mathematical Theory of Computation: Papers in Honour of John McCarthy.* Academic Press, 1991.

[Kowalski and Toni, 1996] R. Kowalski and F. Toni. Abstract argumentation. *Journal of Artificial Intelligence and Law*, 4:275–296, 1996.

[Kowalski, 1974] Robert Kowalski. Predicate logic as programming language. In Jack L. Rosenfeld, editor, *Information Processing 74, Proceedings of IFIP congress 74*, pages 569–574, Stockholm, Sweden, 1974. North-Holland.

[Krause P and J, 1995] Elvang-Goransson M Krause P, Ambler S and Fox J. A logic of argumentation for reasoning under uncertainty. *Computational Intelligence*, 11(1):113–131, 1995.

[Krause P, 1993] Judson P Krause P, Fox J. An argumentation based approach to risk assessment. *IMA Journal of Mathematics Applied to Business and Industry*, 5:249–263, 1993.

[Lamport, 1993] Leslie Lamport. How to write a proof. Technical Report 94, Digital Equipment Corporation, Systems Research Center, 1993.

[Lin and Shoham, 1989] Fangzhen Lin and Yoav Shoham. Argument systems: a uniform basis for nonmonotonic reasoning. In R.J. Brachman, H.J. Levesque, and R. Reiter, editors, *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning*, pages 245–255, Sam Mateo(California), 1989. Morgan Kaufmann.

[Loui and Chen, 1992] R. P. Loui and W. Chen. An argument game. Technical Report WU CS TR 92-47, University of Washington, 1992.

[Loui, 1994] R. P. Loui. An argument and arbitration game (precis). In *Proc. AAAI Workshop on Computational Dialectics*, pages 72–83, Seattle, 1994.

[McCarthy and Hayes, 1969] J. McCarthy and P. J. Hayes. Some philosophical problems from the stnadpoint of artificial intelligence. In B. Meltzer and D. Michie, editors, *Machine Intellignce 4*, pages 463–502. Edinburgh University Press, 1969.

[McCarthy, 1958] John McCarthy. Programs with common sense. In *Proceedings of the Symposium on Mechanisation of Thought Processes*, volume 1, pages 77–84, London, 1958. Her Majesty's Stationery Office.

[McCarthy, 1980] John McCarthy. Circumscription, A form of non-monotonic reasoning. *Artificial Intelligence*, 13:27–39, 1980.

[McDermott, 1982] D. McDermott. Nonmonotonic logic II: Non-monotonic modal theories. *J.ACM*, 29:33–57, 1982.

[Minker and Ruiz, 1993] J. Minker and C. Ruiz. On extended disjunctive logic programs. In J. Komorowski and Z. W. Raś, editors, *Proceedings of the 7th International Symposium on Methodologies for Intelligent Systems (ISMIS'93)*, pages 1–18, Trondheim, Norway, June 1993.

[Moore, 1985] R. C. Moore. Semantical Considerations on Nonmonotonic Logic. *Artificial Intelligence*, 25:75–94, 1985.

[Newton C. A. da Costa, 1995] Otavio A. S. Bueno Newton C. A. da Costa, Jean-Yves Beziau. Aspects of paraconsistent logic. *Bulletin of the Interest Group in Pure and Applied Logics*, 3(4):597–614, 1995.

[Ng *et al.*, 1998a] Benson Hin-Kwong Ng, Kam-Fai Wong, and Boon-Toh Low. A logical framework for reasoning over attacking conflicts in multi-agent systems. In *Proceedings of ECAI'98 Workshop on Conflicts among agents*, Brighton UK, August 25 1998. (to be published).

[Ng *et al.*, 1998b] Benson Hin-Kwong Ng, Kam-Fai Wong, and Boon-Toh Low. Resolving conflicting arguments under uncertainties. To appear in *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, 1998.

[Piotr Rudnicki, 1996] Andrzej Trybulec Piotr Rudnicki. Fixpoints in complete lattices. *Journal of Formalized Mathematics*, 8, 1996.

[Pollock, 1994] John L. Pollock. Justification and defeat. *Artificial Intelligence*, 67:377–408, 1994.

[Pollock, 1995] John L. Pollock. *Cognitive Carpentry.* Bradford/MIT Press, 1995.

[Pollock, 1996] John L. Pollock. Oscar–a general purpose defeasible reasoner. *Journal of Applied Non-Classical Logics*, 6:89–113, 1996.

[Poole, 1988] David Poole. A logical framework for default reasoning. *Artificial Intelligence*, 36(1):27–47, August 1988.

[Prakken and Sartor, 1997] Henry Prakken and G. Sartor. Argument-based logic programming with defeasible priorities. *Journal of Applied Non-classical Logics*, 7:25–75, 1997.

[Prakken and Sergot, 1997] Henry Prakken and M.J. Sergot. Dyadic deontic logic and contrary-to-duty obligations. In D.N. Nute, editor, *Defeasible Deontic Logic*, pages 223–262. Synthese Library, Kluwer, 1997.

[Priest, 1979] Graham Priest. Logic of paradox. *Journal of Philosophical Logic*, 8:219–241, 1979.

[Przymusinski, 1990] Teodor C. Przymusinski. Extended stable semantics for normal and disjunctive programs. In Peter Warren, David H.D.; Szerdei, editor, *Proceedings of the 7th International Conference on Logic Programming (ICLP '90)*, pages 459–480, Jerusalem, June 1990. MIT Press.

[Przymusinski, 1991] Teodor Przymusinski. Stable Semantics for Disjunctive Programs. *New Generation Computing Journal*, 9:401–424, 1991.

[Reed *et al.*, 1996] Chris Reed, Derek Long, and Maria Fox. An architecture for argumentative dialogue planning. In Dov M. Gabbay and Hans Jürgen Ohlbach, editors, *Proceedings of the International Conference on Formal and Applied Practical Reasoning (FAPR-96)*, pages 555–566, Berlin, June 3–7 1996.

[Reiter, 1978] R. Reiter. On closed world data bases. In Inh. Gallaire and J. Minker, editors, *Logic and Data Bases*, pages 55–76, Plenum, New York, 1978.

[Reiter, 1980] R. Reiter. A logic for default reasoning. *Artificial Intelligence*, 13(1–2):81–132, April 1980.

[Rescher, 1977] Nicholas Rescher. *DIALECTICS : A Controversy-Oriented Approach to the Theory of Knowledge.* State University of New York Press, 1977.

[Robinson, 1965] J. A. Robinson. A machine-oriented logic based on the resolution principle. *Journal of the ACM*, 12(1):23–41, January 1965.

[Ross, 1989] Kenneth Ross. The well-founded semantics for disjunctive logic programs. In *Proceedings of the first International Conference on Deductive and Object Oriented Databases, Kyoto, Japan*, pages 1–22, 1989.

[Sakama and Inoue, 1995] Chiaki Sakama and Katsumi Inoue. Paraconsistent stable semantics for extended disjunctive programs. *Journal of Logic and Computation*, 5:265–285, 1995.

[Sakama, 1992] Chiaki Sakama. Extended well-founded semantics for paraconsistent logic programs. In *Proceedings of the International Conference on Fifth Generation Computer Systems (FGCS'92)*, pages 592–599, Ohmsha, 1992.

[Strawson, 1952] P. F. Strawson. *Introduction to Logical Theory.* Methuen, London, 1952.

[Tarski, 1956] Alfred Tarski. *Logic, Semantics, Metamathematics: Papers from 1923 to 1938.* Oxford University, 1956.

[Thomason *et al.*, 1986] Richmond Thomason, Jeff Horty, and D. S. Touretzky. A Calculus for Inheritance in Monotonic Semantic Nets. Research Note CMU-CS 86-138, Carnegie Mellon, 1986.

[Toulmin, 1958] Stephen Edelston Toulmin. *The uses of argument.* Cambridge : University Press, 1958 edition, 1958.

[Verheij, 1996] Bart Verheij. *Rules, Reasons, Arguments - Formal studies of argumentation and defeat.* PhD thesis, Universiteit Maastricht, 1996.

[von Wright, 1951] G. H. von Wright. *An Essay in Modal Logic.* North-Holland Publishing Company, 1951.

[Vreeswijk, 1991] G. Vreeswijk. Abstract argumentation systems. In Michel DeGlas and Dov Gabbay, editors, *Proceedings of the 1st World Conference on the Fundamentals of Artificial Intelligence*, pages 501–510, Paris, France, July 1991. Angkor.

[Vreeswijk, 1997] Gerard A. W. Vreeswijk. Abstract argumentation systems. *Artificial Intelligence*, 90(1-2):225–279, 1997.

[Warren, 1983] David H. D. Warren. An Abstract PROLOG Instruction Set. Technical Report 309, Artificial Intelligence Center, Computer Science and Technology Division, SRI International, Menlo Park, CA, October 1983.

[Witold, 1990] Lukasiewicz Witold. *Non-monotonic Reasoning. Formalization of Commonsense Reasoning.* Ellis Horwood, 1990.

[Wybraniec-Skardowska, 1991] Urszula Wybraniec-Skardowska. *Theory of Language Syntax : Categorical Approach.* Kluwer Academic Pub, 1991.

[Zhang and Foo, 1997] Yan Zhang and Norman Y. Foo. Answer sets for prioritized logic programs. In Jan Małuszyński, editor, *Proceedings of the International Symposium on Logic Programming (ILPS-97)*, pages 69–84, Cambridge, October13–16 1997.

Appendix A

First Oreder Logic (FOL)

Since Frege proposed the preliminary version of first order logic (FOL) in [Frege, 1967], confusion between syntax and semantics have never been clarified until Alfred Tarski. Tarski set up a clean correspondence between syntactical reading ("what we can show") and semantical reading ("what we do mean"). Tarski showed that first order logic enjoys the nice property that there exists a semantical reading, which is lately known as Tarskian semantics, coincides with its syntactical reading, i.e. proof. Tarski shown that the notion of truth in ordinary usages is in semantical level. The correspondence between meaning and proof implies that problem of determining the truth status in ordinary usages can be done on syntactical level.

We briefly describe, a summary of [Davis, 1993], Tarski's formulation of FOL as a prelude to logic programming. FOL uses the following logical symbols: $\neg \rightarrow \wedge \vee \leftrightarrow \exists \forall$; punctuation marks $[](,)\}$ and variables $\psi_1, \psi_2, \ldots$. A vocabulary is 4-tuple $< C, F, R, d >$ where C is a set of constants, F functions, R relations and d is degree of the function and relation. Note that d is a mapping projecting $F \cup R$ to integers i.e. giving the name of function or relation d returns its degree. A term of a vocabulary V is a constant, a variable, or a value of a function $f(\mu_1, \ldots, \mu_n)$ where μ_i is a term of V and $d(f) = n$. An atomic formulas of V is an expression $r(\mu_1, \ldots, \mu_n)$ where $r \in R$ and μ_i is a term. An atomic formula of V is a formula of V and all variables of the formula are free. If A and B are formula of V, $\neg A$, $A \rightarrow B$, $A \wedge B$, $A \vee B$, $A \leftrightarrow B$, $(\exists x)A$ and $(\forall x)A$ are formulas. For formulas $(\exists x)A$ and $(\forall x)A$, variables x in A are bounded. A sentence of V is a formula of V containing no free variables. An interpretation I of V is a 4-tuples $< D, CM, FM, RM >$ where D is the domain, CM the set $\{c_I | c \in C\}$, FM a mapping $f_I : D^n \rightarrow D$ where $f \in F$ and $d(f) = n$, and RM a mapping $r_I : D^n \rightarrow \{0, 1\}$ where $r \in R$ and $d(r) = n$. A valuation of D is a set of elements $\{v_1, \ldots\}$ from D. A valuation of I is a valuation of its domain D. The value of a term ν in a valuation v of I is defined as $Value(\nu, I, v)$ equals to c_I if $\nu \in C$, v_j if ν is variable ψ_j, otherwise $f_I(d_1, \ldots, d_n)$ if $\nu = f(\mu_1, \ldots, \mu_n)$ and $Value(\mu_i, I, v) = d_i$.

Valuation of formula is defined as follow:

- If γ is a formula of V, $Value(\gamma, I, \nu) = r_I(d_1, \ldots, d_n)$ if $\gamma = r(\mu_1, \ldots, \mu_n)$ is an atomic formula and $Value(\mu_i, I, v)$.
- If $\gamma = \neg\alpha$, $Value(\gamma, I, \nu) = 1 - Value(\alpha, I, v)$.
- If $\gamma = (\alpha \rightarrow \beta)$, $Value(\gamma, I, \nu) = 1 - Value(\alpha, I, v) + Value(\beta, I, v)$.
- If $\gamma = \alpha \vee \beta$, $Value(\gamma, I, \nu) = Value(\alpha, I, v) + Value(\beta, I, v)$.
- If $\gamma = \alpha \wedge beta$, $Value(\gamma, I, \nu) = Value(\alpha, I, v) \times Value(\beta, I, v)$.
- If $\gamma = \alpha \leftrightarrow \beta$, $Value(\gamma, I, \nu) = Value(\alpha \rightarrow \beta, I, v) \times Value(\beta \rightarrow \alpha, I, v)$.
- If $\gamma = (\exists\psi_j)\alpha$ $(\gamma = (\forall\psi_j))$, $Value(\gamma, I, \nu) = 1(0)$ if there is a valuation v' of D s.t. $v'_j = v$ and $Value(\alpha, I, v') = 1(0)$ otherwise $Value(\gamma, I, v) = 0(1)$.

Two sentences α and β are truth equivalent if $\alpha \leftrightarrow \beta$. For a sentence γ, $Value(\gamma, I)$ means $Value(\gamma, I, v)$ does not depend at all on v. I is a model of γ, iff $Value(\gamma, I) = 1$. I is a model of a set of sentences S iff I is a model of every sentence of S. For two sentence sets α and A, A entails α, denoted $A \models \alpha$, if model of A is also model of α. A sentence set A is consistent if it has at least one model and otherwise inconsistent.

From the above formulation of Tarskian FOL, it is clear that nothing was mentioned about the feasibility of proofing a thesis. Jacques Herbrand shown that "if a set of first-order logic statements T is inconsistent, there exists a finite grounded subset of it which is inconsistent". Herbrand also introduced the prenex form which is a transformation of a sentence α to α' such that all $(\exists x)$ and $(\forall x)$ symbols are moved to left-hand side and $\alpha \leftrightarrow \alpha'$. Skolem shown that it is possible to removing existential quantifiers $(\exists x)$ in a prenex sentence by instantiating existential variables with value of a Skolem function. Skolem function is nothing more than giving a name to the existed value of variable. As existence of a value may be more than one, Skolen transformation of prenex sentence does not preserve truth equivalence. Interestingly, Skolem shown that Skolem transformation is model preserving in the sense that if a prenex sentence has a model, Skolem transformed of it also has a model. Skolem transformed sentences are lately called clauses with the following form: "$(\forall x_1)(\forall x_2)\ldots(\forall x_n)P$" where P is an atomic formula or an atomic formula preceded by a negation $\neg$ in disjunctive normal form with variables $x_1, x_2, \ldots, x_n$. An atomic formula is in disjunctive normal form if it is in the form of "$l_1 \vee \ldots \vee l_n$" where l_i is an atomic formula. In the following text, we assume any sentences being universal quantified if not specified otherwise. Martin Davis and Hilary Putnam showed that Herbrand's theorem could be very useful in computing models of clauses. As it is possible to show a thesis P is inconsistent with

a theory T in finite time, we can proof a thesis P with respect to T by showing $\neg P$ is inconsistent with T. Davis and Putnam's procedure computes on grounded sentences. Robinson in [Robinson, 1965] showed that Davis-Putnam's procedure could be improved through a technique known as lifting (resolution + unification) which was indeed a rediscovered result of [Herbrand, 1967]. Empirical results showed that Robinson's resolution procedure reduced the search space a lot and could compute far better than Davis-Putnam procedure.

Robert Kowalski was the first one who perceived the implication of Robinson's result. In [Kowalski, 1974], he showed that procedural readings of a clause could be obtained by the truth-equivalence between material implication $A \rightarrow B$ and $\neg A \vee B$. For a sentence $l_1 \vee \ldots \vee l_n$, we define $L^+ = \{l_i | l_i$ is an atomic formula $\}$ and $L^- = \{\alpha_i | l_i$ is an atomic formual α_i preceded with $\neg\}$ then the sentence can be transformed to $\bigwedge_{a \in L^-} a \rightarrow \bigvee_{b \in L^+} b$. L^- and L^+ are called conditions and conclusions of the sentence.

Semantically, a sentence " $\bigwedge_{a \in L^-} a \rightarrow \bigvee_{b \in L^+} b$ " can be read as " $\bigwedge_{a \in L^-} a$ is the preconditions of $\bigvee_{b \in L^+} b$". In procedural way, $\bigwedge_{a \in L^-} a$ must be proved (executed) to complete the proof (execution) of $\bigvee_{b \in L^+} b$. Thus, every clauses have their respective procedural readings.

Appendix B

DAS-I Proof

Before going into the soundness and completeness proof, we shall introduce the following notations to simplify the proof procedure. For an argument Arg,

- tree(Arg) is the argument tree in which Arg wins;
- branch(T) is the set of branches of an argument tree T;
- length(B) is the length of a branch;
- move(B, n) is the nth move of a branch B;
- player(M) is the player that responsible for the move M.

The proofs shown here are in *Lamport* style [Lamport, 1993].

B.1 Monotone proof

Lemma 1 *Given a set of argument S and conflict free subset S_i and $S_i \subseteq S_{i+1}$. If an argument Arg is in $\Pi_S(S_i)$ then Arg is also in $\Pi_S(S_{i+1})$.*

PROOF SKETCH: *We prove the thesis by contradiction. We assume there is an argument Arg in $\Pi_S(S_i)$ but not in $\Pi_S(S_{i+1})$. Using the definition of fix-point iteration, a contradiction is shown.*

ASSUME: *1. $\exists$ argument Arg, s.t. $Arg \in \Pi_S(S_i)$ and $Arg \notin \Pi_S(S_{i+1})$*

⟨1⟩1. *$\exists DArg$ defeating Arg which is not defeated by arguments in S_{i+1}.*

⟨1⟩2. *$S_i \subseteq S_{i+1}$ implies $DArg$ is also not defeated by S_i.*

⟨1⟩3. *$\exists DArg$ defeating Arg which is not defeated by arguments in S_i.*

⟨1⟩4. *By definition, $Arg \in \Pi_S(S_i)$ implies $\nexists DArg$ defeating Arg which is not defeated by arguments in S_i.*

□

Theorem 2 *The fix-point operator* Π *is monotone.*

PROOF:*A direct rephrasing of lemma 1.*

B.2 Soundness proof

Lemma 2 *If arguments Arg is provably justified, then there exists an argument tree such that every move of the proponent in every branch involves only justified arguments.*

PROOF SKETCH: *We prove the lemma by induction on the level of tree(Arg). Let h be the height of tree(Arg). We argue that all proponent moves at level h are justified. By backward induction, we argue that proponent moves at odd level i must also be justified based on justified proponent moves at level* $i + 2$.

ASSUME: *1.* $\exists$ *tree(Arg) s.t. Arg wins all branches.*

2. n = *length(tree(Arg))*

3. $level(Arg, i) = \{m | m \in move(\ branch(\ tree(Arg)\ ),\ i)\ \}$

$\langle 1 \rangle 1$. **Basis :** *at level* $h = n$, $\forall m \in level(Arg, h)$, m *is justified*

$\langle 2 \rangle 1$. *player(m) = proponent*

$\langle 2 \rangle 2$. m *is the last move implies* $\nexists \bar{m}$ *defeating* m

$\langle 2 \rangle 3$. m *is justified by definition*

$\langle 2 \rangle 4$. *Q.E.D.*

$\langle 1 \rangle 2$. **Induction Step :** *If* i *is odd and* $level(Arg, i+2)$ *is justified, then* $level(Arg, i)$ *is also justified*

ASSUME: *1.* $level(Arg, i+2)$ *is justified.*

2. $leaf = \{m | m \in level(Arg, i)$ *and* m *is not the last move of the branch* $\}$

3. $non\text{-}leaf = level(Arg, i)$ *- leaf*

$\langle 2 \rangle 1$. *leaf is justified, by the same reason in* $\langle 1 \rangle 1$.

$\langle 2 \rangle 2$. $\forall m \in$ *non-leaf,* $\exists \bar{m} \in level(Arg, i+1)$ *strictly defeat it.*

$\langle 2 \rangle 3$. *since* $m \in$ *non-leaf and end move of every branch is by proponent, there exists a move* m^* *by proponent after* $\bar{m}$.

$\langle 2 \rangle 4$. m^* *is justified as* m^* *in* $level(Arg, i+2)$.

$\langle 2 \rangle 5$. $\forall \bar{m}$ *of* m, $\bar{m}$ *is strictly defeated.*

$\langle 2 \rangle 6$. m *is justified by fix-point definition.*

$\langle 2 \rangle 7$. *Q.E.D.*

$\langle 1 \rangle 3$. **Completion :** *For all odd integer* $i < n$, $level(Arg, i)$ *is justified. Moves of proponent only occur at odd level of an argument tree.*

□

Theorem 3 *All provably justified arguments are justified.*

PROOF SKETCH: *Every provably justified argument, Arg, has an argument tree in which all proponent moves are justified. Arg is one of those proponent moves and the result follows.*

⟨1⟩1. *By lemma 2,* $\exists t \in tree(Arg)$ *s.t. all proponent moves of* t *are justified.*

⟨1⟩2. *Root of* t *is justified.*

⟨1⟩3. *Arg is root of* t.

⟨1⟩4. *Arg is justified.*

□

B.3 Completeness proof

Lemma 3 *If Arg is a justified argument, then there exists an argument tree such that every moves of the proponent involves only justified arguments.*

PROOF SKETCH: *We start to construct a tree inductively with a justified argument Arg. At every odd level i, we show that there exists justified arguments DArg strictly defeating the defeaters of level i. DArg can then form level* $i+2$*'s move. Then, we can inductively construct an argument tree by non repetitive moves of the opponent.*

ASSUME: *1. Arg is justified.*

2. T is an argument tree.

3. Arg is at level 1 of T.

⟨1⟩1. **Basis :** *Level 3 is justified.*

ASSUME: *1. D is the set of defeaters defeating* $level(Arg, 1)$.

⟨2⟩1. $\forall d \in D, \exists \bar{d} \in$ *fix-point strictly defeating d.*

⟨2⟩2. $DArg = \{\bar{d} | d \in D\}$ *is justified by definition.*

⟨2⟩3. *let* $level(Arg, 3) = DArg$

⟨2⟩4. *Q.E.D.*

⟨1⟩2. **Induction Step :** *If level i is justified, there exists justified* $level(Arg, i+2)$ *defeating* $level(Arg, i+1)$

ASSUME: *1.* $level(Arg, i)$ *is justified.*

2. $level(Arg, i+1)$ *is non-empty.*

3. $arguable = \{m | m \in level(Arg, i) \cap m$ *is defeated by* $level(Arg, i+1)\}$

⟨2⟩1. $level(Arg, i+1)$ *defeats* $level(Arg, i)$

⟨2⟩2. $\forall m \in arguable$, *there exists* $\bar{m}$ *in fix-point defeating* $level(Arg, i+1)$

.

⟨2⟩3. $level(Arg, i+2) = \{\bar{m} | m \in arguable\}$

⟨2⟩4. $level(Arg, i+2$ *is justified and defeats* $level(Arg, i+1)$

⟨2⟩5. *Q.E.D.*

⟨1⟩3. **Completion :** *Thus, there exists a proof tree in which every proponent level involves only justified arguments.*

□

Theorem 4 *All justified arguments are provable.*

PROOF SKETCH: *We prove the thesis by contradiction. Start with there exists a justified argument Arg that is not provable. Using lemma 3 we find an argument tree to support Arg and lead to a contradiction.*

ASSUME: *1. Arg is justified.*

2. By Lemma 3 and the hypothesis, $\exists T$, an argument tree such that

a. every move of a proponent is justified

b. a proponent cannot move in a branch B

⟨1⟩1. *Let*

1. m = move(B,length(B))

2. $\bar{m}$ = move(B,length(B)-1)

⟨1⟩2. *$\bar{m}$ is justified.*

⟨1⟩3. *$\exists m'$, s.t. m' strictly defeats m.*

⟨1⟩4. *m' is a viable move for the proponent.*

□

Appendix C

Sherlock Holmes' *Silver Blaze* Excerpts

C.1 Double life

...”Undoubtedly. But in examining his belongings I was fortunate enough to discover not only the method of the crime but even its motives. As a man of the world, Colonel, you know that men do not carry other people's bills about in their pockets. We have most of us quite enough to do to settle our own. I at once concluded that Straker was leading a double life and keeping a second establishment. The nature of the bill showed that there was a lady in the case, and one who had expensive tastes. Liberal as you are with your servants, one can hardly expect that they can buy twenty-guinea walking dresses for their ladies. I questioned Mrs. Straker as to the dress without her knowing it, and, having satisfied myself that it had never reached her, I made a note of the milliner's address and felt that by calling there with Straker's photograph I could easily dispose of the mythical Derbyshire.···

C.2 Poison stable boy

··· ”Undoubtedly. He has neither a knife nor any sign of a wound. The evidence against him is certainly very strong. He had a great interest in the disappearance of the favourite. He lies under suspicion of having poisoned the stable-boy; he was undoubtedly out in the storm; he was armed with a heavy stick, and his cravat was found in the dead man's hand. I really think we have enough to go before a jury.”

Holmes shook his head. ”A clever counsel would tear it all to rags,” said he. ”Why should he take the horse out of the stable? If he wished to injure it, why could he not do it there? Has a duplicate key been found in his possession? What chemist sold him the powdered opium? Above all, where could he, a stranger to the district, hide a

horse, and such a horse as this? What is his own explanation as to the paper which he wished the maid to give to the stable-boy?" ···

Supplement

1 Evaluation Questions

1. (the archetypal even loop)
   ```
   p <- not q
   q <- not p
   ```

2. (reasoning by cases)
   ```
    <- not q
   q <- not p
   r <- p
   r <- q
   ```

3. (stratification)
   ```
   p <- not q
   q <- not r
   ```

4. (tautological loop)
   ```
   p <- not q
   q <- q
   ```

5. (the archetypal odd loop)
   ```
   p <- not p
   ```

6. (conditional odd loop)
   ```
   p <- q, not p
   ```

7. (3-loop)
   ```
   p <- not q
   q <- not r
   r <- not p
   ```

8. (odd loop + even loop)
   ```
   r <- not r
   r <- q
   p <- not q
   q <- not p
   ```

9. (odd loop + even loop; inefficient version)
   ```
   r <- not r
   r <- not p
   p <- not q
   q <- not p
   ```

10. (conditional odd loop + even loop)
    ```
    r <- p, not r
    p <- not q
    ```

```
    q <- not p
```

11. (2&3-loop)
```
    p <- not q
    q <- not r
    r <- not p, not q
```

12. (query as an integrity constraint)
```
    r <- q
    r <- t
    t <- s
    q <- not p
    s <- not p
    p <- not t
    t <- not p
    <- not r      %% This is the query.
```

13. (the law of exclusive middle)
```
    q <- p
    p | -p <-
```

14. (incoherent even loop)
```
    q <- not p
    p <- not q
    q <- p
    p <- q
```

15. (inclusive disjunction)
```
    p | q <-
    q <- p
    p <- q
```

16. (minimality)
```
    p | q <-
    p <- q
```

17. (disjunctive + integrity constraint)
```
    p | q <-
    p <- q
    <- not q
```

18. (normal default)
```
    flies(X) <- bird(X), not -flies(X)
    -flies(X) <- penguin(X)
    bird(X) <- penguin(X)
    bird(polly) <-
    penguin(tweety) <-
```

19. (naming defaults; another formalization)

```
flies(X) <- bird(X), -ab(X)
-flies(X) <- penguin(X)
bird(X) <- penguin(X)
bird(polly) <-
penguin(tweety) <-
-ab(X) <- bird(X), not ab(X)
ab(X) <- -flies(X)
```

20. (bad formalization for Nixon diamond)

```
dove(X) <- quaker(X), not ab_quaker(X)
hawk(X) <- republician(X), not ab_republician(X)
quaker(nixon) <-
republician(nixon) <-
<- dove(X), hawk(X)
```

21. (correct formalization for Nixon diamond)

```
dove(X) <- quaker(X), not ab_quaker(X)
hawk(X) <- republician(X), not ab_republician(X)
quaker(nixon) <-
republician(nixon) <-
ab_quaker(X) <- hawk(X)
ab_republician(X) <- dove(X)
```

22. (barber's "non-"paradox)

```
shaves(X,Y) <- barber(X), citizen(Y), not shaves(Y,Y)
shaves(X,X) <- barber(X)
citizen(X) <- barber(X)
barber(noel) <-
citizen(casanova) <-
```

23. (incoherent closed world assumption)

```
<- -p(a), -p(b)
-p(X) <- dom(X), not p(X)
dom(a) <-
dom(b) <-
```

24. (modified closed world assumption)

```
<- -p(a), -p(b)
<- not -p(a), not -p(b)
-p(X) <- dom(X), not p(X)
p(X) <- dom(X), not -p(X)
dom(a) <-
dom(b) <-
```

25. (disjunctive + closed world assumption)

```
p(a) | p(b) <-
-p(X) <- dom(X), not p(X)
dom(a) <-
dom(b) <-
```

```
26. (naming CWA)
    q <- p
    q <- -p
    -q <-
    -p <- assume(-p), not p
    assume(-p) <- not -assume(-p)
    -assume(-p) <- not assume(-p)

27. (nonmonotonic ATMS)
    p <- b
    q <- a, not p
    p <- not q
    a <- not out_a     %% Either "a" is assumed
    out_a <- not a     %%     or "a" is not assumed.
    b <- not out_b     %% Either "b" is assumed
    out_b <- not b     %%     or "b" is not assumed.

28. (abduction)
    p <- b
    q <- a
    <- q, b
    <- not q, not b
    a <- not -a
    -a <- not a
    b <- not -b
    -b <- not b
    <- not q           %% This is the query.

29. (definition of terminal nodes)
    -terminal(X) <- arc(X,Y)
    terminal(X) <- node(X), not -terminal(X)
    node(a) <-
    node(b) <-
    node(c) <-
    node(d) <-
    node(e) <-
    arc(a,b) <-
    arc(b,c) <-
    arc(c,c) <-
    arc(b,d) <-

30. (story of employees)
    adequate_income(X) <- employed(X,Y)
    no_income(X) <- person(X), not adequate_income(X)
    -employed(X,Y) <- person(X), company(Y), not employed(X,Y)
    person(ryuzo) <-
    person(noriko) <-
    person(pootaro) <-
```

```
company(icot) <-
company(ntt) <-
employed(ryuzo,ntt) | employed(ryuzo,icot) <-
employed(noriko,icot) <-
```

31. (story of university students)

```
eligible(X) <- highGPA(X)
eligible(X) <- minority(X), fairGPA(X)
-eligible(X) <- -fairGPA(X)
interview(X) <- student(X), not eligible(X), not -eligible(X)
student(taro) <-
student(hanako) <-
student(manabu) <-
student(keiko) <-
minority(taro) <-
fairGPA(taro) <-
-fairGPA(hanako) <-
highGPA(manabu) <-
highGPA(keiko) | fairGPA(keiko) <-
```

32. (Yale shooting; bad formalization)

```
holds(P,result(A,S)) <- holds(P,S), next_action(A,S), not ab(P,A,S)
holds(alive,s0) <-
holds(loaded,result(load,S)) <-
holds(dead,result(shoot,S)) <- holds(loaded,S)
ab(alive,shoot,S) <- holds(loaded,S)
next_action(load,s0) <-
next_action(wait,result(load,s0)) <-
next_action(shoot,result(wait,result(load,s0))) <-
answer(P) <- holds(P,result(shoot,result(wait,result(load,s0))))
```

33. (meeting-scheduling)

```
person(a) <-
person(b) <-
day(mon) <-
day(tue) <-
room(212) <-
room(213) <-
reserved(212,mon) <-
reserved(213,tue) <-
busy(a,agent,mon) <-
busy(b,self,tue) <-
cand(X,self,Y) <- person(X), day(Y), not busy(X,self,Y)
cand(X,agent,Y) <- busy(X,self,Y), not busy(X,agent,Y)
openroom(Room,Day) <- day(Day), room(Room), not reserved(Room,Day)
selves_meeting <- holdmeeting(Day,Room,self,self)
holdmeeting(Day,Room,self,self) <-
    cand(a,self,Day), cand(b,self,Day), openroom(Room,Day)
holdmeeting(Day,Room,Ida,Idb) <-
```

```
        cand(a,Ida,Day), cand(b,Idb,Day), openroom(Room,Day),
        not selves_meeting

34. (analogy)
    apartfrom(ball,block) <-
    apartfrom(planet,sun) <-
    apartfrom(electron,neucleus) <-
    attracts(X,Y) <- astroheavy(X), object(Y)
    attracts(X,Y) <- poselect(X), negelect(Y)
    attracts(X,Y) <- negelect(X), poselect(Y)
    astroheavy(sun) <-
    object(planet) <-
    poselect(neucleus) <-
    negelect(electron) <-
    revolves(planet,sun) <-
    <- revolves(ball,block)
    revolves(X,Y) <- not contra, appli, apartfrom(X,Y), attracts(Y,X)
    appli <- apartfrom(X,Y), attracts(Y,X), revolves(X,Y)
    contra <- apartfrom(X,Y), attracts(Y,X), not revolves(X,Y)
%%-------------------------------------------------------------------------
```

2 Answer

```
%%-------------------------------------------------------------------
%% Answers (not verified completely) by Katsumi Inoue
%%
 1. {p}, {q}
 2. {p,r}, {q,r}
 3. {q}
 4. {p}
 5. none
 6. {}
 7. none
 8. {q,r}
 9. {q,r}
10. {q}
11. {q}
12. {q,r,s,t}
13. {p,q}, {~p}
14. none
15. {p,q}
16. {p}
17. none
18. {b(p),p(t),b(t),~f(t),f(p)}
19. {b(p),p(t),b(t),~f(t),f(p),ab(t),~ab(p)}
20. none
21. {quaker(nixon),republician(nixon),dove(nixon),ab_republician(nixon)},
```

```
    {quaker(nixon),republician(nixon),hawk(nixon),ab_quaker(nixon)}
22. {barber(jun},citizen(ken),citizen(jun),shaves(jun,jun),shaves(jun,ken)}
23. none
24. {~p(a),p(b)}, {p(a),~p(b)}
25. {~p(a),p(b)}, {p(a),~p(b)}
26. {~q,~assume_neg_p}
27. {out_a,out_b,p}, {a,out_b,q}, {a,out_b,p}, {out_a,b,p}, {a,b,p}
28. {a,~b,q}
29. {node(a),node(b),node(c),node(d),node(e),
     arc(a,b),arc(b,c),arc(c,c),arc(b,d),
     ~terminal(a),~terminal(b),~terminal(c),terminal(d),terminal(e)}
30. {person(ryuzo),person(noriko),person(pootaro),company(icot),company(ntt),
     employed(ryuzo,icot),employed(noriko,icot),
     ~employed(ryuzo,ntt),~employed(noriko,ntt),
     ~employed(pootaro,icot),~employed(pootaro,ntt),
     adequate_income(ryuzo),adequate_income(noriko),no_income(pootaro)},
    {person(ryuzo),person(noriko),person(pootaro),company(icot),company(ntt),
     employed(ryuzo,ntt),employed(noriko,icot),
     ~employed(ryuzo,icot),~employed(noriko,ntt),
     ~employed(pootaro,icot),~employed(pootaro,ntt),
     adequate_income(ryuzo),adequate_income(noriko),no_income(pootaro)}
31. {student(taro),student(hanako),student(manabu),student(keiko),
     minority(taro),fairGPA(taro),~fairGPA(hanako),highGPA(manabu),
     eligible(manabu),eligible(taro),~eligible(hanako),
     highGPA(keiko),eligible(keiko)},
    {student(taro),student(hanako),student(manabu),student(keiko),
     minority(taro),fairGPA(taro),~fairGPA(hanako),highGPA(manabu),
     eligible(manabu),eligible(taro),~eligible(hanako),
     fairGPA(keiko),interview(keiko)}

%%-------------------------------------------------------------------

32. (Yale shooting (2)--modified; 1991.10.24)
    holds(P,result(A,S)) <- holds(P,S), next_action(A,S), not ab(P,A,S)
    holds(alive,s0) <-
    holds(loaded,result(load,S)) <- next_action(load,S)
    holds(dead,result(shoot,S)) <- holds(loaded,S)
    ab(alive,shoot,S) <- holds(loaded,S)
    next_action(load,s0) <-
    next_action(wait,result(load,s0)) <-
    next_action(shoot,result(wait,result(load,s0))) <-
    answer(P) <- holds(P,result(shoot,result(wait,result(load,s0))))

32. (Answer)
    {next_action(load,s0),
     next_action(wait,result(load,s0)),
     next_action(shoot,result(wait,result(load,s0))),
     holds(alive,s0),
     holds(loaded,result(load,s0)),
```

```
holds(dead,result(shoot,result(load,s0))),
ab(alive,shoot,result(shoot,result(load,s0))),
holds(alive,result(load,s0)),
holds(alive,result(wait,result(load,s0))),
holds(loaded,result(wait,result(load,s0))),
holds(dead,result(shoot,result(wait,result(load,s0)))),
ab(alive,shoot,result(wait,result(load,s0))),
holds(loaded,result(shoot,result(wait,result(load,s0)))),
answer(dead),answer(loaded)}
```

%%----------

32. (Yale shooting (3)--a la Gelfond & Lifschitz; 1995.10.27)

```
holds(alive,s0).
-holds(loaded,s0).

holds(loaded,result(load,S)) <- situation(S).
-holds(loaded,result(shoot,S)) <- situation(S).
-holds(alive,result(shoot,S)) <- holds(loaded,S).

holds(loaded,S) <- -holds(alive,result(shoot,S)), holds(alive,S).
-holds(loaded,S) <- holds(alive,result(shoot,S)).

holds(F,result(A,S)) <- holds(F,S), action(A), not ab(F,A,S).
-holds(F,result(A,S)) <- -holds(F,S), action(A), not ab(F,A,S).

holds(F,S) <- holds(F,result(A,S)), not ab(F,A,S).
-holds(F,S) <- -holds(F,result(A,S)), not ab(F,A,S).

ab(loaded,load,S) <- situation(S).
ab(alive,shoot,S) <- situation(S), not -holds(loaded,S).
ab(loaded,shoot,S) <- situation(S).

action(load).
action(shoot).
action(wait).

sitiuation(s0).
situation(result(A,S)) <- action(A), situation(S).
```

%%----------